.EXE

MAY 1990 VOL 4 ISSUE 11

£3.20

The Software Developers' Magazine

Is the OOP Silver Bullet made of brass? Pascal inventor Niklaus Wirth says so.

How it all began. We find out where OOP came from, and where it is going.

Programming transputers in occam and C.

A new trick to tap the power of the 80386 from MS-DOS.

Don't get ripped off! There's an 01 phone number converter inside.

THE TRAINING LEADERS

Applications and Systems Programmers

What comes after C?

C++ is fast emerging as the next step forward for today's C developer.

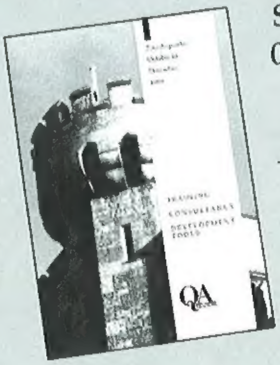
QA's C++ courses are specially designed for you to make the most effective transition to C++ in the minimum time. They combine excellent course materials and tutorials with practical hands-on experience. Covering not just the syntax of the language, but also the concepts of object oriented design.

And this is just part of QA's comprehensive C training.

A wide range of courses for the beginner, right through to the highly experienced systems programmer, ensure total training to the exacting standards required for project success.

In fact, there's no better way than QA to keep up-to-date with current and future C based developments.

So for further details make sure you post the coupon today. Or simply call Samantha Trinder on 0285-655888.



'C' COURSE OPTIONS

C Primer
C Programming
C Advanced Software Development
C Standards
C Techniques for OS/2 and PM
C Techniques for DOS
C++
C++ Techniques for PM and Windows – Common View

ALSO CONSULTANCY

C Coding Standards
C Tooling Standards
C or C++ based design

Microsoft Authorised

Glockenspiel APPROVED

Please complete and mail to QA Training, Cecily Hill Castle, Cirencester, Glos GL7 2EF.

Please send me details on QA's □ C courses □ C++ courses □ C consultancy

Please send me details on QA's courses in: □ OS/2 and PM □ Windows □ Systems Support □ Networking □ DOS

NAME TITLE

COMPANY

ADDRESS

POSTCODE

TEL. NO.

QA TRAINING

CIRCLE NO. 880

Window

Help

.EXE

Vol 4 Issue 11

May 1990

Editor
Robert Schifreen

Deputy Editor
Will Watts

Editorial Assistant
Wendie Leek

Display Advertising Manager
Sandra Inniss-Palmer

Group Recruitment Sales Manager
Jane Lyons

Recruitment Sales Executives
Jeremy Cox
Ed Butcher

Reader Services & Promotions
Helena Adams

Design & Layout
Katerina Adams
Raichel Affleck

Office Administrator
Carla Pearce

Cover Photography
Richard Cooke

Publisher
Jon Howell

Editorial & Advertising: 081-994 6477
Subscription Enquiries: 044 282 4501

Editorial Policy
We aim to provide news, product reviews and technical features for those who develop PC software for both commercial sale and internal company use. Our policy is not to review any software product until it is available in its final form, in order to provide accurate figures on code size and speed. The Magazine welcomes articles from readers - please ask for our contributors' guide.

Subscriptions
.EXE is a magazine for managers and programmers involved in the development of software for the IBM PC and compatibles. .EXE is available only on subscription, which costs £35 per year (11 issues) in the UK. The Magazine is published on the 26th of the month preceeding the cover date. There is no January issue. International subscription rates can be found on the subscription card in this issue. Back issues are available at £3.50 each. To subscribe, call 044282 4501. We can invoice your company or take VISA or ACCESS cards.

.EXE Magazine is independent and not affiliated to any vendor of hardware, software, or services. Published by Process Communications Ltd, 10 Barley Mow Passage, London W4 4PH, England. Telephone: 081 994 6477 Fax: 081 994 1533 ISSN:0268-6872

Reacting to Readers
Our annual Reader Survey gives us an insight into the tools, methods, language and hardware that is being used, and that will be used in the process of software development.

We aim to keep in touch with software developers as much as possible. For this reason, we hold twice-yearly meetings at the .EXE offices and around the country, where readers can talk directly to the editorial staff and suggest ways in which the Magazine could be improved.

Not GUI enough

Apart from when I am giving it an occasional treat, my PC runs in text mode. The applications that I use - a word processor, plus a myriad programmer's utilities, sampled in the line of duty - are all character based. To link an object module, I type 'LINK PROG;'. I do not pick up the object file icon with the mouse and drag it over to the linker icon (which, by the way, is a picture of a short length of chain, in the bitmap of my imagination). As a non-GUI PC user, I am not unusual. For example, if you, as a .EXE reader, have had any OS/2 experience, then you may count yourself as one of the privileged few - a recent survey suggested that you are outnumbered several times by your UNIX brethren.

Now this is something of a disappointment. It is five years since I first saw a GUI running on a PC. By 1990, said the magazine editorials of 1985, we shall all be utter WIMPs: forever resizing windows and double-clicking like good 'uns. (It was always 'WIMP' in those days, 'GUI' seemed to come much later; perhaps because buzzwords, like lithium batteries, wear out after a few years, and must be replaced with an equivalent.) Our keyboards, said the pundits, will shrivel and drop off from lack of use. And at the time, to a naive *Tomorrow's World* believer like myself, this seemed almost as plausible as cheap, safe nuclear power. Indications of the problems to come - such as the turgid response of the old Apricot F1 running GEM, where the machine seemed to have barely enough power to animate the mouse cursor - were cheerfully ignored.

The reasons for the failure of GUI on the PC - for failure is what it is - are various. I have already mentioned performance. Mac users may not mind the slowness of bit-mapped graphics, but they are not able to make a direct comparison with a character-based application. Try talking to a Real User who has been 'upgraded' from, say, Lotus 1-2-3 to Microsoft Excel, but is still working on the same machine; and see how well-pleased he is with the switch. The search for swift response times may lead to poor software design, but as a saleable feature, it still beats alleged compatibility with vapourware into a cocked hat.

There is also the matter of creating the software. You don't have to be Adele Goldberg to knock out a program that lets the operator specify options using a row of 'check boxes', but I bet you could produce a menu-driven program, driven by the keys '0' - '5' in a fraction of the time. No doubt the general introduction of OOP languages will narrow the gap. Notice, incidentally, that, once again, this problem has long since been overcome on the Mac.

Are we still waiting for the right implementation? Digital Research's GEM does not seem a realistic option, unless you plan to port the DTP package that you have written to the Atari ST range. Windows, although it has enjoyed more success recently, and is about to be upgraded again, has always had a slight smell of death to it: Windows is, after all, sitting on top of MS-DOS, and it has been drilled into us that MS-DOS is doomed. The OS/2 V2.0 requires 6 *(six!)* MB of RAM to run; your software will need to be spectacularly good to justify your customer's hardware investment.

And so comes the dismal realisation. It's not that GUI hasn't really happened for PC-class machines; it is the feeling that it is never going to happen; we are always going to have factions of users backing diverse systems, with the majority using text-based systems. Any window (sorry about that) of opportunity for creating a *de facto* PC standard has pased. I'm a fan of GUIs, and feel sure that, eventually, all interactive software on small machines will be written this way, for the same reason that cars now all share the same arrangement of three pedal controls, despite historic difficulties of standardisation. I'm worried that, on the car that I drive now, they are still quarrelling about whether to put the throttle on the left or in the middle. *WRW*

THE C LANGUAGE

Microsoft C V6 is a complete rewrite with improved optimisation and a new Programmer's Workbench High C V1.6 has been considerably improved, with better Microsoft C compatibility, and new documentation

C COMPILERS

Avocet C	PC-DOS	£210
Aztec C86 Professional	MS-DOS	£120
Aztec C86 Developer	MS-DOS	£175
Aztec C86 Commercial	MS-DOS	£290
CI C86 Plus v1.3	MS-DOS	£290
ECO-C88 V4.0	MS-DOS	£ 70
HIGH C (Metaware) v1.6	MS-DOS	£420
HIGH C 386 v1.6	PL386&MS-DOS	£635
Hi-Tech C	MS-DOS	£135
Lattice C v6	OS/2&MS-DOS	£175
Mark Williams LETS C v4	MS-DOS	£ 60
Microsoft C v6	OS/2&PC-DOS	£269
Microsoft QuickC v2	PC-DOS	£ 60
M'soft QuickC/Quickasm	PC-DOS	£105
MIX C	MS-DOS	£ 20
MIX Power C & lib s'ce	PC-DOS	£ 29
Turbo C v2	PC-DOS	£ 79
Turbo C Professional v2	MS-DOS	£160
Topspeed C for OS/2	OS/2	£335
Topspeed C (std)	PC-DOS	£135
Topspeed C Professional	PC-DOS	£270
Watcom Express C v7	PC-DOS	£ 65
Watcom v7 opt comp + tls	PC-DOS	£250
Watcom C/386	PL386&MS-DOS	£625
Zortech C v3	PC-DOS	£ 45
Zortech C++ v2	PC-DOS	£120
Z'tech C++ v2 Developer	PC-DOS	£270
Aztec C86 Developer	CP/M-86	£245
Aztec C86 Personal	CP/M-86	£140
Hi-Tech C	CP/M-86	£135
Lattice C	CDOS	£395
HIGH C (Metaware)	Flex OS 286	£420
Avocet C	CP/M-80	£210
Aztec C Personal 1.06D	CP/M-80	£120
Aztec Commercial 1.06D	CP/M-80	£200
Eco-C w. Lib s'ce	Z80 + CP/M-80	£ 60
Hi-Tech C	Z80 + CP/M-80	£ 99
Mix C	Z80 + CP/M-80	£ 35
Toolworks C/80 v3.1	CP/M-80	£ 45
Aztec C65 V1.05	Apple DOS	£200
Aztec C65 V3.20	Apple PRO-DOS	£245
Aztec C68K MPW C	MACINTOSH	£110
Aztec C68K C	MACINTOSH	£ 75
Aztec C68K/AM Developer	AMIGA	£175
Aztec C68K/AM Prof'snl	AMIGA	£120
Aztec C68K/ST Developer	ATARI	£175
Aztec C68K/ST Prof'snl	ATARI	£120
Hi-Tech C	ATARI	£ 99
Lattice C v3.04	ATARI	£ 85
Mark Williams C v2	ATARI	£110
Laser C	ATARI	£135
Prospero C	ATARI	£105

C CROSS COMPILERS

We supply 2500AD, Avocet, Aztec, Lattice, IAR and Hi-Tech Cross Compilers hosted on MS-DOS and targeted on Z80, 6502, 6801, 68HC11, 6301, 6809, 7811, 8051, 8096, 68000 & 68020. Please call for information or advice.

C INTERPRETERS

Instant C/16M supports large programs on 286/386 machines.

C-terp V3.0X	PC-DOS	£165
Interactive C	PC-DOS	£195
Introducing C	PC-DOS	£ 85
Living C Plus	PC-DOS	£135
Instant-C v3	PC-DOS	£380
Instant-C/16M	PC-DOS	£680
RUN/C Professional	MS-DOS	£110

DEBUGGING TOOLS

Logitech Multiscope	OS/2	£220
MagicCV	386&PC-DOS	£210
Periscope I/512K	PC-DOS	£580
Periscope II	PC-DOS	£115
Periscope II-X	PC-DOS	£ 95
Quaid Analyser	PC-DOS	£125
Soft-Ice	386&PC-DOS	£390
Soft-Ice + Magic CV	386&PC-DOS	£535

DISK COPYING SERVICE

We can copy files to and from 600 disk formats including CP/M, CP/M-86, MS- DOS, PC-DOS, ISIS, APPLE, SIRIUS, BBC, TORCH, APRICOT, HP-150, TRSDOS, DEC RT- 11, IBM BEF, ATARI ST, AMSTRAD, MACINTOSH.

Our charge is £10.00 + disk + VAT with discounts on small quantities and disks are normally despatched within 24hrs of receipt. For more information call us.

C LIBRARIES

DATABASE

Btrieve	OS/2	£400
Btrieve v5	MS-DOS	£170
Btrieve/Network	MS-DOS	£400
XQL	OS/2,PC-DOS	£540
Xtrieve Plus	OS/2,PC-DOS	£400
C/Database T'lchest sce	PC-DOS	£ 39
CBTREE	(source any C)	£135
C-Index/Plus	(source any C)	£175
C-ISAM (L,MS)	MS-DOS	£220
C-tree	(source any C)	£230
Essential Btree (s'ce)	PC-DOS	£120
R-tree	MS-DOS	£170
db-VISTA III s'gle user	PC-DOS	£575
db-Query v2 s'gle user	PC-DOS	£575
Lattice dBCIII (L,MS)	MS-DOS	£165
Lattice dBCIII+	OS/2&MS-DOS	£385
SftFcus Btree&Isam (sce any C)		£ 80
Virtual Memory File Man.	PC-DOS	£ 80
Zortech Btree/ISAM	PC-DOS	£ 75

GRAPHICS

Enhanced Graphics Tlkt	PC-DOS	£170
Essential Graphics v3	PC-DOS	£235
GFX Graphics	PC-DOS	£ 75
GraphiC v5 (CI,L,DS,MS)	PC-DOS	£290
Graphics Menu (MSC,TC)	PC-DOS	£105
GSS Kernel system	OS2,PC-DOS	£350
GSS Graph Dev T'kit	OS2,PC-DOS	£350
GSS Graphic Plotting sys	PC-DOS	£245
HALO v3 (MSC5,LAT,TC)	PC-DOS	£230
HGraph (MSC)	PC-DOS	£ 80
Ingraf (MSC)	PC-DOS	£175
MetaWINDOW (CI,L,MS,T)	PC-DOS	£145
MetaWINDOW/Plus	PC-DOS	£190
MetaWINDOW/Premium	PC-DOS	£315
TurboWINDOW/C (Turbo)	PC-DOS	£ 79
QuickWINDOW/C (QC)	PC-DOS	£ 79

SCREEN & WINDOWS

Vitamin C (MS5.1)	OS/2&PC-DOS	£240
Power Screen (MS&TC)	PC-DOS	£105
Blaise View Mngr.(s'ce)	PC-DOS	£245
Lattice Curses	PC-DOS	£105
Entelekon Windows(s'ce)	PC-DOS	£105
Multi-windows (MS,L)	PC-DOS	£190
Panel Plus II (source)	PC-DOS	£270
Vitamin C (source)	PC-DOS	£165
Vermont Views (MSC,TC)	PC-DOS	£345
Greenleaf Datawindow	PC-DOS	£180
Greenleaf Makeform	PC-DOS	£ 60
C-Scape(+s'ce&Look&Feel)	PC-DOS	£330
Zortech Windows(ZC++ZC3)	PC-DOS	£ 39
Curses/PC w. s'ce	PC-DOS	£265

GENERAL FUNCTIONS

C+O Class Lib (MS,TC)	PC-DOS	£155
C+O Cl's Lb (MS)	WINDOWS&PC-DOS	£155
C/Utilities T'chest sce	PC-DOS	£ 39
CQL s'ce	PC-DOS	£270
C-Worthy Int.Lib (L,MS)	MS-DOS	£175
Code Runner	PC-DOS	£105
Faircom Toolbox Prof	PC-DOS	£665
Greenleaf Functions s'ce	PC-DOS	£105
G'leaf Super Functions	PC-DOS	£135
G'leaf Business Mathlib	PC-DOS	£180
The Heap Expander(MS&TC)	PC-DOS	£ 65
Smorgasbord C-food	PC-DOS	£ 65
C Tools Plus/6.0 (MS)	PC-DOS	£105
Power Search	PC-DOS	£105
Turbo C Tools v2	PC-DOS	£105
Essential C Utils s'ce	PC-DOS	£145
Entelekon Funct(source)	PC-DOS	£105
Pro-C & Workbench	PC-DOS	£300
Resident-C (MS,L)(s'ce)	PC-DOS	£145
Zortech Supertext	PC-DOS	£ 39

COMMS LIBRARIES

C Asynch Manager V3s'ce	PC-DOS	£135
Essential Comms(L,MS,T)	PC-DOS	£195
Greenleaf Comms(source)	PC-DOS	£135
Multi Comms (L,MS)	PC-DOS	£190
Silvercomm C Asynch	PC-DOS	£145
Zortech Comms (QC&TC)	PC-DOS	£ 65
Zortech Comms(ZC++&ZC3)	PC-DOS	£ 65

SCIENTIFIC LIBRARIES

Wiley Scientif.Lib.v2	ANY C	£175
Mathpak 87 (L,MS)	MS-DOS	£ 85

PROGRAMMERS UTILITIES

PC-Lint	OS/2,MS-DOS	£ 85
C-Documentor	PC-DOS	£195
C-Scan	PC-DOS	£195
Clear+ for C	PC-DOS	£165
Lattice Comp Companion	MS-DOS	£ 65

REAL TIME & MULTI-TASKING TOOLS

Concurrent C (PC/MPX)sce	MS-DOS	£ 60
Multi C (L,MS,CI,TC)	PC-DOS	£190
Op.Sys.T'box (MS) s'ce	PC-DOS	£ 65
Timeslicer v3.01 (L)	PC-DOS	£175
Timeslicer v5 (MS)	PC-DOS	£175
Over-C (L,MS)	PC-DOS	£225

PROGRAMMING TOOLS

Ada Compilers	Algol Compilers
Assemblers & Libs	AWK
Basic Compilers	Basic Interpreters
Basic Utilities	Basic Libraries
BCPL Compilers	C Compilers
C Interpreters	C Libraries
C Utilities	Cobol Compilers
Comms.Libraries	Cross Assemblers
Database Libs.	Debuggers
Dis-assemblers	Editors
Engineers Libs.	Expert Systems
Forth	Fortran Compilers
Fortran Libraries	Graphics Libraries
Icon	Linkers/Locaters
Lisp	Modula-2
Nial Interpreters	OPS 5
Pascal Compilers	Pascal Libraries
Prolog	Rexx
Risc	Screen Libraries
Smalltalk	Snobol

We stock many items for which there is no space in these advertisements.

PRICES & DELIVERY

Prices do not include VAT or other local taxes but do include delivery in UK and Europe. Please check prices at time of order, ads are prepared some weeks before publication.

This page lists some of our products. Call us for a complete pricelist.

Order by phone with your credit card.

ASSEMBLERS

MS Macro-86 v5.1	OS/2&PC-DOS	£ 95
PharLap 386 ASM.LINK	MS-DOS	£400
PharLap 386 ASM/Linkloc	MS-DOS	£565
PharLap 386/VMM	MS-DOS	£240
Optasm	MS-DOS	£ 95
Turbo Assembler/Debugger	PC-DOS	£105
2500AD Z80 ASM	CP/M-80	£ 70
Dig.Res. RMAC	CP/M-80	£180
SLR Z80ASM	CP/M-80	£ 40
SLR Z80ASM-PLUS	CP/M-80	£140
SLR MAC	CP/M-80	£ 40
SLR MAC-PLUS	CP/M-80	£140
SLR 180 (Hitachi)	CP/M-80	£ 40
SLR 180-PLUS (Hitachi)	CP/M-80	£140

Not all assemblers are supplied with a linker. Check before ordering.

PROLOG LANGUAGE

Arity Standard	PC-DOS	£ 79
Arity Prolog Int. v5.1	PC-DOS	£225
Arity Int. & Comp. v5.1	PC-DOS	£495
Flex Expert Dev Sys	PC-DOS	£1210
LPA PROLOG Prof v3	PC-DOS	£755
Micro-PROLOG v3.1	MS-DOS	£ 70
Prolog-86 plus v1.0	MS-DOS	£195
Prolog-2 Personal	PC-DOS	£145
Prolog-2 Programmer	PC-DOS	£495
Prolog-2 Professional	PC-DOS	£995
Turbo-Prolog v2	PC-DOS	£ 79
ADA Educ.Prolog	MS-DOS	£ 26
ADA FS Prolog	MS-DOS	£ 45
ADA VMI Prolog	MS-DOS	£ 70
ADA VML Prolog	MS-DOS	£150
Micro-Prolog v3.1	CP/M-86	£ 70
Micro-Prolog v3.1	CP/M-80	£ 70
Prolog-1 v2.2	CP/M-80	£225
LPA Mac Prolog v3	Apple Mac Pls	£450

GREY MATTER

2 Prigg Meadow, Ashburton, Devon TQ13 7DF

TEL. (0364) 53499

GREY MATTER

2 Prigg Meadow, Ashburton, Devon TQ13 7DF

TEL. (0364) 53499

GREY MATTER

2 Prigg Meadow, Ashburton, Devon TQ13 7DF

TEL. (0364) 53499

CIRCLE NO. 881

News

Microsoft C V6.0

Microsoft has launched, and is now shipping, Microsoft C V6.0 Professional Development System. Bad news first: it's an enhanced version of V5.1, which provides neither 32-bit code generation nor OOP support. Windows and OS/2 development still requires the purchase of extra tools. The optimiser, however, has been totally rewritten, and Microsoft's figures say that the average compiled program will now be 10% smaller and 7-10% faster. Optimisation can now be controlled by embedded pragmas. There is a tiny memory model for producing .COM files - an unexpected lurch towards Zortech compatibility. New language features include the introduction of an alternative fast calling convention (arguments are passed in registers instead of on the stack) and a new based pointer type. These are 16-bit pointers which are interpreted as offsets to a segment specified in the declaration. It's best explained with a fragment of code:

```
_segment seg;
char _based(seg) *string;
char _far *farstring;
/*....*/
/* Get a based heap */
seg = _bheapseg(HEAPSIZE);
/* Get space for string */
string = _bmalloc(seg, STRLEN);
_fstrcpy((char far*)string, farstring);
```

Note the use of the new reserved words `_segment` and `_based` in the declarations, and that the based pointer is usually treated as a far pointer. However, the compiler can make extra optimisations with the based type when performing pointer arithmetic. Personally, I will need some convincing that this justifies further complicating the mess which is pointer handling in 80x86 C compilers.QuickC is no longer bundled, but there's a Quick Compilation switch on the main compiler, which turns off the optimiser. (QuickC itself has been upgraded to version 2.5, and is sold separately.) There's also an incremental compilation switch, which restricts compilation to the parts of a program which have changed - but this can only be used if the Quick Compilation switch is also set. A new inline assembler lets you embed assembly language within C source code. Microsoft is claiming ANSI Standard compatibility for the product, except that double-byte characters are not supported.

The Programmer's Workbench is a WIMP-based front end from which you can call the compiler, linker etc. There is also a source code browser, which shows you the hierarchy of procedure calls, plus the various interdependencies. The Workbench can record the compilation switch settings for each module, and you can have different settings for, say, the release version and the development version of a project.

The CodeView debugger is now up to version 3, and boasts a new interface that fits in with the Workbench. There's still no facility for remote debugging using two machines and a serial port, but CodeView will now sit in extended or expanded memory, allowing you to debug larger programs. RRP £335.

Financial library

Greenleaf Software Inc, well-known for its range of C libraries running under MS-DOS and OS/2, has produced a new version of its Financial MathLib (previously called 'Business MathLib') package. The software consists of a suite of arithmetic functions based on the proprietary 'DEC' number representation system, which uses 11 bytes to store numbers of up to 18 digits of accuracy, with a floating decimal point. Using DEC numbers, the programmer is able to maintain much greater control over rounding errors and other inaccuracies which traditional floating-point number systems introduce into code. The DEC type is supported by trigonometric, statistical and financial functions (such as loan amortisation and depreciation), as well as the more straightforward arithmetic, I/O and type conversion routines.

Greenleaf says that the new version of the library runs up to five times as fast as the previous release. Financial MathLib is priced at $395 US, which includes the source code. It is compatible with the Microsoft, Borland, Zortech and Lattice C compilers. For more information, contact the manufacturer on 0101 214 248 2561.

Pi not irrational

PiEdit is a programmer's editor which comes in three versions: MS-DOS (£195), OS/2 (£249) and UNIX/XENIX (£349). As you may guess from these prices, it does rather more than EDLIN. There is support for multiple files in multiple resizable windows, a macro language, automatic backups of current files after a given number of keystrokes, syntax checking for C, C++, dBASE, Pascal and a few other languages, background printing, complete 'customisability' and an off-line/batch editing capability. The feature that is being pushed hardest, though, is PiEdit's 'undo' facility. The program maintains a 32 KB buffer of your keystrokes, which you can undo *selectively* - so it is possible to correct an error without losing subsequent edits. PiEdit is distributed in the UK by IDS (071 631 0548).

On Form

APForm is an add-in program for Borland's Paradox 3. Run from within the Paradox environment, the utility allows the easy creation of both forms and reports with complex lines and boxes. Priced at £59, it is distributed by Dunstan Thomas Ltd (0705 822254). The company also supplies a range of other Paradox development tools.

PC Check

PC Check is a PC diagnostic program aimed at OEMs and system resellers. It is a software package which tests all the facilities of a PC compatible computer. After determining the configuration and performing tests on the RAM (including the extended and expanded flavours), disks, display adapter, keyboard and even the printer, the program produces a diagnostic report. The package comes from Eurosoft Ltd (0202 297315).

SoftPC

Insignia Solutions' SoftPC package runs on the Macintosh, making it into a PC/XT emulator. Now the company has introduced an add-on module, which lets the Mac emulate an AT with EGA graphics. The emulation is incomplete, though; it doesn't do 80286 protected mode and can't handle OS/2. The software is priced at £299 for the base SoftPC package and £149 for the EGA/AT add-in module. Insignia is on 0494 459426.

Fast 486

Intel may not be producing any 33 MHz 80486 chips yet, but why should a little thing like that stop anybody from producing a 33 MHz 486-based machine? AMT's MicroFrame 433 has these specifications; it is based on a hand-screened CPU (nominal clock rate: 25 MHz) and costs about £8000 for a system with 140 MB hard disk and colour VGA. AMT says that it is the world's first; talk to the company on 081 450 3222.

Windows Prolog

Quintec has released a version of its Prolog compiler for the Microsoft Windows/386 environment. According to the manufacturer, the product makes extensive use of the Windows facilities, providing support for graphics, icons, menus and so on. A single copy of the package costs £1800. Quintec's phone number is 0865 791565.

DataComms Book

VNU (071 4394242) has published the fourth edition of The DataComms Book, a comprehensive directory of data and telecommunications. the book costs £57 including UK p&p.

PUT THEM

Such is the confidence we have in Wordtech's dBXL and Quicksilver products that we'll even let you put them to the test. Right now, we're offering you the chance of a free demo disk on both products whilst stocks last.

dBXL is a fully dBASE-compatible database management system but with improved performance, sophisticated user interface and a number of language extensions. Quicker and easier to use, its price/performance makes it hard to beat.

2

Quicksilver, a dBASE language compiler, offers the software developer a balance between dBASE compatibility, performance, ease of use and extended language features. This makes Quicksilver an indispensable programming tool and the perfect addition to dBXL.

dBXL and Quicksilver provide:

true windowing • automatic memory variables
multi-dimensional arrays • desktop publishing hooks
user defined functions • valid & help
database graphing • soundex
EMS support • multi-tasking
error correction • .PCX stored images

Contact Software Limited now on 081-743 5000 (South) or 0472 361558 (North) or complete the coupon. Alternatively, if you're already convinced call us for details of your nearest dealer.

CIRCLE NO. 882

THE TEST

I would like to put both dBXL and Quicksilver to the test.
Please send me a free demo disk. □

Please send me details of my nearest dealer □

WORDTECH SYSTEMS

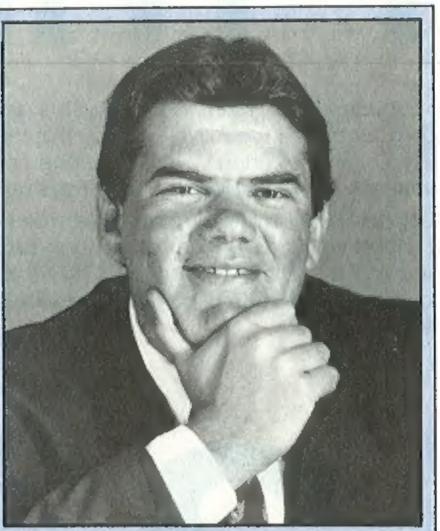
SOFTWARE *sells Hardware*

Name ....
Company..
Address ....
....
....
.... Tel No ....

Software Limited, Software House, Unit 7/8 Acton Park Industrial Estate, The Vale, London W3 7SY

Kahn not cant

Philippe Kahn, President of Borland, was in London recently, to address a packed audience of students and lecturers at the Computer Science Faculty of Imperial College. His chosen subject was Object Oriented Programming. Kahn compared the current vogue for OOP with the similar fashion for AI in the early 1980s. Since that time, instead of taking over the world, the commercial application of AI has been increasingly confined, so that now it is used only in niche markets. Kahn claimed that the vital difference between the technologies - the thing that would make OOP take off - is the fact that some OOP languages are supersets of existing mainstream languages. This despite the imperfections of the underlying languages; he described C as 'PDP-11 assembler'. He predicted that, by 1999, the standard teaching language would be Pascal with objects, the most widely used development system would be C++, and that President Dan Quayle would have endorsed Object BASIC.



Concerning the more immediate future, I asked Mr Kahn about imminent release of Borland's own Turbo C++ V1.0 (expected late May). In a recent interview, Bjarne Stroustrup himself had expressed worries about this product, because Borland had not purchased a copy of AT&T's V2.0 language specification. Mr Kahn promised that the Borland implementation would be completely compatible with V2.0, and would include a full set of class libraries (Zortech's rival product omitted V2.0 stream libraries), so presumably the company has acquired a second-hand copy of the spec.

The release of the Turbo Debugger V2.0 package, which we featured in March '90 issue, is to be held back to coincide with the release of Turbo C++.

Phonebase

In September, British Telecom's Directory Enquiries service goes on-line. You'll be able to access the system with a modem, and you'll have access to the same system currently used by the operators when you call the service the normal way. If you want to try out the system now, and have access to a 1200/75 baud modem, call 0800 919199. That's the Phonebase service desk, who will issue you with a temporary user id and password over the phone, and will send you an application form for a permanent one. There's no charge for using the service, apart from the phone call itself, which is charged at 'b' rate, which is around 5p per minute.

C's conscience

All C programmers have heard of UNIX's `lint` utility, so-named (according to apocryphal legend) because it extracts bits of fluff from C code. I've used Gimpel Software's PC-lint product in the past, and think that it deserves a different metaphor. PC-lint is the programmer's conscience - it's very hard and tiresome to do its bidding, but if you don't, you will eventually meet your Day of Judgement.

This is a preface to the news that PC-lint has now reached V4.0. There are 70 new reproachful diagnostic messages that the program can generate as it busybodies its way through your code. I think my favourite, which will save you time rather than pick up mistakes, is the message that warns of redundant `#include` files. There are 28 so-called Elective Note messages, which draw each and every cast (explicit and implicit) to your attention. This is fussy even by PC-lint's standards - it's intended as a porting aid - which is recognised by the fact that these messages default to *off*.

Gimpel Software always includes a new 'spot the bug' challenge with each subsequent release of PC-lint. Here is the latest offering:

```
#if PROTOTYPES
double sqrt(double);
#else
double sqrt();
#endif

main()
{
  printf("sqrt(2) = %g\n", sqrt(2));
}
```

Anybody who is forced to type this in and run it to find out what is wrong is utterly weak and feeble, and should be struck off the Holy Role of C Programmers (I would say that, because I have got the crib). PC-lint V4.0 is available directly from Gimpel in the US (0101 215 584 4261), priced $139.

X User Group

You will have to move fast to get to the European X User Group's Spring Conference, to be held in Cambridge on 4th May, so this item serves mainly to alert interested parties to the existence of EXUG (phone: 0954 211860), and to make an appeal to conference organisers, who would like to be mentioned in .EXE's news pages, to send us the information at least two months in advance. We have had a lot of matter relating to April events come in recently, when we were nearly past the deadline for May.

Lattice RPG

Lattice has released V3.1 of its RPG II Development System. The new version has an enlarged data area, and also supports dynamic link libraries, so it is now possible to port larger System 3X programs down to the PC. Unusually, the compiler itself is protected, following heavy pirating of previous releases. Lattice's UK distributor is Roundhill Systems. Last time I mentioned them, I produced an out-of-date number, so my apologies and the correct number: 0672 84535.

R without K

They have an impressive list of speakers at the UKUUG conference, to be held in London 11th-13th July, including one Dennis Ritchie (yes, that Dennis Ritchie) who will be talking about C, and Rob Pike, also from AT&T Bell Labs, whose subject will be the planned successor to the UNIX OS, currently dubbed 'Plan 9'. You don't have to be a UKUUG member to register, but it will cost you around £300. Details on 0763 73039.

Menu Works

If you shepherd a flock of ignorant end users, who need protecting from the rigours of the MS-DOS environment, you might consider a £34.50 program called Menu Works. The program installs itself and produces a hierarchy of menus, based on the applications that it finds already present on the machine. You can customise these menus, for example, password protecting certain applications, and there is even a simple programming language. Menu Works is available from BCS Ltd (081 308 0291).

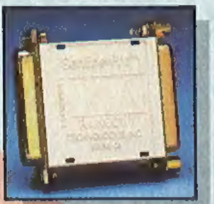
Graphic DESQview

Quarterdeck has announced a graphical version of its MS-DOS multi-tasker DESQview. The new package will be based on X Windows, and will allow PC class machines to run DOS and X tasks simultaneously. The software is scheduled to ship in August.

SOFTWARE SECURITY

With £1 Billion Worth Of Protected Software...

SentinelPro™



• Runs under DOS, OS/2 and Xenix • Algorithm technique (Never a fixed response) • External parallel port installation • Minimal implementation effort • Higher level language interfaces included • 100 times faster than fixed-response devices (1 ms) • ASIC design for reliability

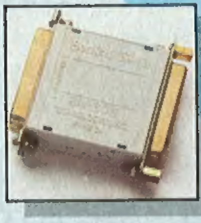
Sentinel-C™

• Protects multiple packages with one device • 126 bytes of non-volatile memory programmed before shipment of the software • Rainbow supplies a unique adapter for programming the unit • Higher level language interfaces included • Runs under DOS, OS/2 and Xenix • External parallel port installation

Eve™

• For the Macintosh SE and II • Complies with Apple Desktop Bus Interface requirements • Rainbow-assigned developer passwords to prevent tampering by other developers or sophisticated "hackers" • 7 locks per key, usable individually or in combination, on one or up to seven applications

SentinelShell™



• Runs under DOS on IBM PCs and compatibles • Protects without requiring access to the source code • Completely transparent to the end user • User-friendly software • Pocket-size key attaches quickly to any standard PC parallel port • ASIC design for reliability

DataSentry™

• Completely user-installable • Pocket-sized external device • Menu-driven, user-friendly interface • Single- or multi-user security system • Audit trail, log-on identifiers and automatic encryption/decryption of entire directories • Secures data transmitted by modems • Prevents recovery of data by utility programs

Rainbow Is The Safe Bet.

Software developers creating the latest applications for the IBM PC/XT/AT, PS/2 and compatible systems can now turn to the Software Sentinel range of hardware keys for the *first-class* in *world-class* software protection.

There's the best-selling SentinelPro, known worldwide for its virtually unbreakable security, its ASIC technology and its invisible operation.

A close relation, the Sentinel-C for custom configurations, enables multiple package protection with a single device.

For the Apple market, security-minded Mac software developers can now secure their return-on-investment, too. Eve plugs into the Mac ADB connector and is completely transparent to the user–providing up to seven programmable security locks per key.

Rainbow's latest protection strategy is the SentinelShell–that lets users place a "shell" around existing, off-the-shelf programs. Access can be limited to those issued a key, so that libraries, educational establishments and corporations can very simply guard their investments.

Available soon from Rainbow is the DataSentry, a user-installed key that provides low cost security for sensitive data in both database applications and corporate/banking environments.

No matter where you sell your software worldwide, stay in control of your distribution and revenue by choosing the internationally accepted standard in protection... Rainbow Technologies. Be sure. Protect your pot of gold at the end of the rainbow.

CALL TODAY OR ORDER NOW.

□ Please send me a SentinelPro Evaluation Kit.
I enclose £50 + VAT Payable to Rainbow Technologies Ltd.
□ Please debit my credit card. Access ___ Visa ___ Amex ___
Expiry Date __________ (PLEASE TICK)

□ Please send me more information.
Name ______________________
Position ______________________
Address ______________________
Telephone ______________________
Signature ______________________ EXE 5/90
RAINBOW TECHNOLOGIES LTD., Shirley Lodge, 470 London Road,
Slough, Berkshire SL3 8QY.

RAINBOW TECHNOLOGIES
Shirley Lodge, 470 London Road
Slough, Berkshire SL3 8QY
Tel: 0753-41512 Fax: 0753-43610

CIRCLE NO. 883

Don't Blink

Nantucket's Clipper compiler is notorious for generating memory-inefficient code, and a series of products has been released which address this problem. The latest is BLINKER, a dynamic overlay linker, which is produced by the gloriously named US company Blink Inc. The manufacturers claim that BLINKER is six to ten times faster than rivals PLINK86plus and .RTLink, and, because it can automatically create overlay units, it cuts the amount of memory required to run applications by up to 50%. Other features include transparent memory defragmentation, serial number encryption, the ability to 'burn-in' Clipper environmental variables (and so avoid the need for lines like 'SET CLIPPER = F50' in your customer's AUTOEXEC) and the suppression of an application's attempts to use expanded memory - apparently necessary because of a bug in Clipper's LIM handler.

BLINKER costs £189 from its UK distributor QBS Ltd, whose telephone number is 081 994 6477.

CompuServe in Europe

CompuServe is an on-line mail and file transfer system, like CIX and BIX. After 10 years in business in the US, it has 550,000 subscribers, which makes it the largest system of its type. Now the company has launched CompuServe Forum, a collection of access points that encourage European use of the system. Before now, access from outside the US has required expensive international phone calls, or use of the International PSS network.

The system currently has around 900 databases, where subscribers discuss computing, programming and a lot of non-technical subjects too. There are also several text databases, and direct contact with technical support people from various hardware and software companies.

Access from any European country currently costs $12.50 per hour (everything's still in dollars) for speeds up to 2400 bps. From the UK, access is via the Istel network, which is an extra $9.50 per hour, which is around £13 in total . This is around half the price of using the international PSS routes. Details from 0800 289378.

Pecan on UNIX

Pecan's Pascal compiler is based on the UCSD P-code system. The compiler produces special pseudo object code ('P-code'); this is interpreted to the native machine-code equivalent at run time. There is also an integrated editor and file system, which sit 'on top' of the host operating system, making its operation very machine-independent. It was first devised in the early days of Pascal to minimise porting effort, and so speed up the spread of the language to new platforms.

This being the case, it is surprising that, until now, Pecan Pascal has not been available for SCO UNIX. The compiler costs £1200 ex VAT, and is available directly from the manufacturer in Bristol (phone 0272 425012).

Boot card

US-based Award Software has produced a PC card which can boot up a machine by itself. The card is based on a ROMmed version of DR DOS, and is primarily intended for diskless workstations, freeing them from the need to boot off the network. US price is $199, the company's phone number is 0101 408 370 7979.

Very Sharp

The PC-6220, Sharp Electronics's newly-announced portable, sounds like a masterpiece of miniaturisation. With dimensions of 8.5" x 11" x 1.4", and weighing 6 lbs with battery, the machine has a 12 MHz 80286 CPU, 20 MB hard disk, 1MB RAM and a VGA screen based on colour LCD technology. The only thing missing is a floppy disk drive. The machine should ship in June, when price details will become available.

Giants merge

Lotus, king of the spreadsheets, is planning to merge with Novell, the network company. The resulting giant would become the largest microcomputer software company in the world - ahead of Microsoft itself. The merger is expected to go through in July.

RISC mainframe

Thame Microsystems (0844 261456) has announced a range of 88000-based workstations. The Opus Personal Mainframe range runs UNIX on its RISC processor - and MS-DOS on a secondary 80x86 processor, which also handles I/O to the main CPU. The machine also has AT compatible expansion slots, so the workstation can use cheap peripherals.

Panel Plus

Panel Plus is a well-reputed screen management library for C compilers. The newly released V2.1 of the package has improved CUA conformance, support for the Microsoft C Professional Development System V6.0 (including OS/2 DLLs), Watcom and Metaware 386 compilers and a QuickHelp-compatible help database of the routines. The price, which includes source code, is £295, the supplier is Roundhill (0672 84535).

EC okays Pirates

The European Commission proposes to legalise the decompilation of commercial software - this is considered an infringement of copyright in many countries. At a recent conference of leading computer and legal experts, representing all major EC countries, the consensus was that this 'could endanger the development of the industry'.

Better BASIC

Microsoft's Product Manager for BASIC, Todd Neilson, flew in from the US last month, and .EXE had the chance to talk to him. We asked about BASIC V7.0 (the Professional Development System, which we're hoping to review soon), and about the future for all Microsoft languages. Some of the more important things he told us were:

- The launch of BASIC V7.0 doesn't mean the death of QuickBASIC. QB will continue to be enhanced, using 'hand-me-down' improvements from BASIC PDS. The awkward situation where QB had a number of features that were not present in BASIC V6.0 will not happen again.
- Microsoft is planning a new BASIC product, which will be object-oriented and graphics-based. A command-line switch will allow code generation for Windows or OS/2 Presentation Manager (this high-level way of writing graphical apps will come to all Microsoft languages in time).
- Stub files. Microsoft acknowledges that stub files are no substitute for a well-written library package and intelligent linker.The embedded ISAM system will go multi-user in the next release. Also, the next release of Microsoft's SQL server will include the necessary libraries to allow SQL statements to be embedded in BASIC programs.
- Microsoft acknowledge that big arrays are not really possible under BC7. True, you can now break the 64 KB limit with arrays, but you are still stumped if your array is larger than 128 KB. This will be fixed in a future version.
- Microsoft are looking at ways of making assembly language more accessible from BASIC. Future versions of the language may well have an integrated assembler, as does the latest version of QuickC.
- In QuickBASIC, there are two different code generators. One is invoked when you run a program inside the QB environment, and the other generates .EXE files. In BC7, this is no longer the case. There is only one code generator.

See us at NETWORKS '90 Stand No. 2

IPX or Netbios networks examined through remote controlled PCs

NetOp from Richmond Systems offers fast and easy troubleshooting for the network manager.

Thanks to a new software concept, it is now possible to monitor and control multiple PCs on a local area network either from a single workstation on that network or remotely using modem links.

NetOp combines the following valuable features.

- *Uses only 1 to 2 kbyte of memory on each remote station*
- *Full graphics support including EGA and VGA*
- *Multiwindowing management facility*
- *Password and keyboard lock facilities*
- *Operates in conjunction with established remote communications packages*

This simple-to-install package has an international reputation and has already proved its value throughout Europe and Scandinavia.

To find out more about NetOp and details of your local dealer please contact:

Richmond Systems Ltd
3, Church Terrace,
Richmond, Surrey
TW10 6SE

Tel: 081-940 6986
Fax: 081-948 2026

NetOp

CIRCLE NO. 884

Letters

We welcome short letters on any subject that is relevant to software development. Please write to The Editor, .EXE Magazine, 10 Barley Mow Passage, Chiswick, London W4 4PH. Unless your letter is marked 'Not for Publication', it will be considered for inclusion on this page.

Dear .EXE,

I can see the committee just rising from their seats, whacking each other on the back, breaking out the port or sherry or whatever they keep locked away in the Red Conference Room, congratulating themselves on having at last defined the new standard FORTRAN 88. A nice touch that; giving it a name that as much as says,'Hey, we're two years late and we're admitting it!', as I suppose a sort of ironic way of implying; 'well hell, just look at what we've done!' (But then two years late for anything connected with FORTRAN is good going, I guess.)

Having scanned your article on this new breakthrough, this New Dawn in the evolution of software, it struck me that if you want a sneak preview of FORTRAN 99 you couldn't do much better than to read *Pascal: User Manual & Report* by Jensen and Wirth (circa 1976).

But die-hard FORTRAN programmers needn't be too alarmed: although productivity and source maintainability can be improved by the use of languages like Pascal, Modula-2, FORTRAN 99 and so on, it is still possible - with a bit of ingenuity - to create a flowchart-style spaghetti of a program requiring much debugging, and making its creator indispensable to his employer as the guru of the machine (an old GEC mainframe with any luck).

I can't wait to hear what's in store for COBOL...

Everard Cunion
IIS Limited
Langley, Slough

Dear Sir,

Further to the article 'If I may interrupt' (March '90 issue), I would like to take issue with Mr Margolis's statement concerning the difference in speed between two systems which are communicating with each other via UARTs.

He states that 'any difference in speed between two systems will become cumulative, causing errors'. This is not so, because the UART is, of course, an asynchronous device designed to cope with the slightly different clock speeds between the transmitter and receiver. The UART re-synchronises at the beginning of the start bit and samples each following bit at its centre point. Assuming eight data bits and one parity bit, there can be a speed difference of 0.5 bit in 10.5 bits (4.76%) before timing errors will occur. This is a massive tolerance compared with the accuracy of the crystals used as the clock source for the UARTs. If two stop-bits are sent and the receiving end is set to look for both, then the tolerance is reduced to 0.5 bit in 11.5 bits (4.35%).

R A Stevenson
County Durham

Dear Mr Schifreen,

I have followed, with some interest, the recent debate regarding ANSI.SYS in your magazine. I would like to recommend a short TSR called ANSI.COM written by Michael J Mefford of Ziff Communications. This is a public domain utility, freely available (with source) from a number of UK bulletin boards, which is not only faster and more efficient than ANSI.SYS, but offers a number of significant enhancements. The adjustable key assignment buffer can be cleared or redefined, the ANSI mapping can be enabled or disabled at will, and the whole program can be readily uninstalled. This is an excellent replacement for ANSI.SYS and, to date, I have not heard of any problems with it.

Use of this program solves many of the problems and objections to the conventional ANSI device driver approach.

Chris Roper
COGITAIRE Limited
Devon

Dear Sir,

My name is Doru Turturea. I am a Senior Software Engineer at the Research Institute of Computers of Bucharest, which is the capital of Romania. I am also the General Secretary of the Romanian Computer Science Society. We have been able to form this society as a consequence of our People's Revolution.

At the Research Institute for Computers, we lost many valuable books, software packages and machines in the Revolution. Please could you publish this appeal of help to your readers: we desperately need books, magazines on C and C++ languages, diskettes, user-interfaces, generators, expert systems shells, CASE tools and PC-compatible machines to run them on. We are going to a market economy, with free enterprise, but we desperately need help, and we should be very grateful if you could send any of the above products.

Doru Turturea
Senior Software Engineer
Casuta Postala 2 -122
Sector 1
Bucuresti 71200
Romania

Dear Mr Schifreen,

I would like to point out that the product SuperTest, referred to on page 6 of your March issue, is the result of the co-operation between two companies: ACE Associated Computer Experts bv of Amsterdam, the Netherlands and HCR Corporation of Toronto, Canada. We would be pleased if you would print a rectification, informing your readers of this joint transatlantic relationship.

Marco Roodzant
ACE Associated Computer Experts bv
Amsterdam, The Netherlands

Dear Robert,

In his article on formal methods of software development (.EXE Vol 4, Issue 7), Darrel Ince writes: 'Some specialised software tools known as theorem provers have been developed...' Are any of these tools commercially available?

John Naylor
Cambridge University Press
Cambridge

Darrel Ince replies:
Mr Naylor has identified the Achilles heel of formal methods. The systems that I described currently exist only as prototypes at Manchester University and Imperial College, London. However, there is a commercial Pascal verifier, called the Stanford Pascal Verifier, which ia available from Stanford University, US.

EXE

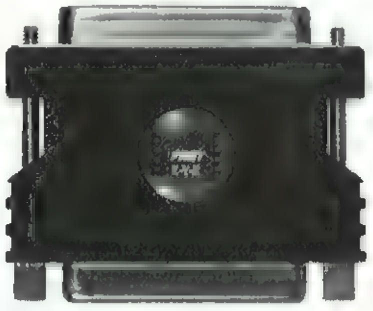
5 out of 5 hackers prefer other software protection methods to Hardlock E-Y-E®

What hackers dislike...
Hardlock E-Y-E was designed using cryptographic principles. It took the experience and know-how of Germany's No. 1 in software protection and the leading edge technology of a US semiconductor company to create the ultimate software protection tool. Hardlock E-Y-E is based on a custom chip featuring secure algorithmic response rather than simple bit swapping or counting schemes.

What software developers like...
Hardlock E-Y-E combines all the features software developers require in a single product: algorithmic response to provide security and an optional non-volatile memory to allow custom configurations. FAST Electronic has made implementation of Hardlock E-Y-E in your software easy. Use HL-Crypt to protect .EXE or .COM files, or incorporate high level language interface routines in your software. The algorithm parameters and the contents of the memory can be programmed in seconds using our Crypto-Programmer card. This unique card guarantees that no one else can burn your original codes. Simply plug the card into any PC slot and start up your own Hardlock E-Y-E workshop.

What your customers will like...
Hardlock E-Y-E allows unlimited backup copies. The device is shipped with the software for the user simply to plug into the parallel interface and forget. Daisy chainability, outstanding reliability (no battery is needed), and the most compact High-Tech design ensure that your customer will accept Hardlock E-Y-E.

What your accountant will like...
Hardlock E-Y-E needs no factory coding. This ensures optimum delivery schedules and stock flexibility. Revenues will go up as software piracy and multiple usage are prevented. Despite its wealth of features, Hardlock E-Y-E's prices remain competitive.



Hardlock E-Y-E
programmable, algorithmic response and memory option – all in one.

...As more and more software developers, customers and accountants appreciate the Hardlock E-Y-E device, hackers like it less and less.

FAST

Fast Electronic GmbH

Order your demo unit today. Contact Magnifeye,
235-239 Walmer Road, Walmer Studio #6, W11 4EY, Telephone 071 221 8024, Fax 071 792 3449.

CIRCLE NO. 885

OOP meets Modula-2

The OOP programming solution is being liberally applied to all kinds software. Niklaus Wirth dares to doubt the universality of this silver bullet.

Professor Niklaus Wirth is one of the most important and influential figures in the field of Computer Science. He began his academic career at the Swiss Federal Institute of Technology (the Eidgenössische Technische Hochschule, usually abbreviated to ETH). He went on to study in Canada and the USA, receiving a PhD from the University of California in 1963. After interludes working at the Navy Electronics Laboratory in San Diego, and teaching at Stanford University, he returned to Switzerland in 1967. Later on, in 1976, he spent a sabbatical year at the famous Xerox PARC in Palo Alto. He still works at ETH.

Professor Wirth is probably best known for creating a series of programming languages: Euler (1963, his dissertion project), ALGOL W (1966), Pascal (1971), Modula (1975), Modula-2 (1980) and Oberon (1988). Each these languages incorporate certain design principles, such as simplicity and abstraction, which are now generally considered (due, in no small part, to Wirth's efforts) as 'good design'.

It is a sad fact that our field of Computer Science is overly dominated by fads. These appear in times of acute difficulties, are praised as powerful medicine against the major ills, and are carried by high hopes of all in despair. In the area of software and programming, the often-cited 'software crisis', first openly acknowledged in 1968, made *structured programming* popular. It was an expression of the recognition that complex software can only be understood if it is orderly and structured. The development of huge systems, programmed by armies of 'analysts', made it evident that co-ordination, in the form of documented interface specifications observed by all participating workers, were mandatory. Management became a dominant topic, and all these aspects were somehow covered by the new wave called *software engineering*, implying the claim of professional approach.

The most recent slogan is *object-oriented programming*. It expresses a different view of systems, focused on decentralised control, and oriented in the area of systems programming. Every such trend has its legitimate reasons and goals, and it is appropriate to investigate its usefulness for one's particular objectives. Such a study is necessary in order not to succumb to the negative aspects of a fad: to apply it where inappropriate simply out of fear of being called old-fashioned. It is vital to understand the issues and foundations of a new discipline. Otherwise we will not master the discipline, but be mastered by it.

What is O-O?

At the core of the object-oriented concept is, I believe, decentralised control. The prime example, which is well-suited to explain the idea, is an operating system. A conventional system contains a central routine, which accepts input from a keyboard and dispatches control to the routine specified for interpretation of the command. An

Figure 1 - Conventional vs Object-Orientated Terminology

type	class
variable	object, instance
procedure	method
call	message
extension	inheritance

even simpler example is the 'operating system' of a desk calculator, which selects the routine according to the function key pressed. Modern workstations, with their multiple window (viewer) capability, require a more sophisticated approach. Typically, operations are demanded through a mouse click. The action to be taken depends on the position of the displayed cursor. It is initially unknown to the system, and depends on the type of viewer in which the cursor happens to be located. Each viewer is considered to carry its own mode of command interpretation; in short, each is regarded as an *object with its own behaviour.* This scheme is implemented by executing a search for the descriptor representing the viewer designated by the current cursor position, and then by dispatching control to a routine assigned to that descriptor, a so-called *handler.* Naturally, different (types of) views must be able to contain different handlers. Instead of control being centralised in a single dispatcher (in which the identities of the destinations are explicitly specified), control is distributed among the handlers, whose identity and number is not specified in the dispatcher's program text.

Incidentally, this view of a structure, with its own routines for interpreting its data, coincides with the notion of the abstract data type. The type declaration does not only specify the types and structure of data, but also the applicable operators and function. Variables are said to be instances of the type. In the community of object-oriented programmers, the set of objects with identical data structure and handler is called a *class,* and an object is an instance of a class, in the same way that a variable is an instance of a type.

It is often desirable to be able to *derive* a new class. This is where instances of some new class share the properties (ie attributes and operations) of objects of an existing class, but have extra properties. These new instances become special members of the original class and form a subclass. A typical example is given by subclasses of viewers. Text viewers, graphic viewers, picture viewers will share all properties of viewers, but will also have additional operators suitable for handling texts, graphics or pictures.

Some programmers find it attractive to view computer systems like humans. An object-oriented system is then compared with a human society. A symptom of this anthropomorphic view - which I can find misleading rather than useful - is the notion that a subclass *inherits* the properties of its superclass. Thus the subject of inheritance has found its entry into the programmer's technical jargon. It might be added in passing, that the term 'subject-oriented' would have been more consistent with the popular anthropomorphic view than 'object-oriented'. After all, in the conventional sense, it is the subject that displays a characteristic behaviour and receives messages. The object, on the other hand, plays only a passive role.

It is by no means accidental that the paradigm of object-oriented programming - we bow to convention and adopt the misnomer - originated in the application area of simulation of systems with discrete events. There emerged the need to represent abstractions of agents with properties and behaviour. Such abstractions were first expressed in languages Simula-1 [1] and Simula-67 [2]. The main focus still lay on simulating the collective and concurrent actions of classes of agents, using an interpreter with a single processor. The idea of using processes or, more precisely, coroutines, remained intimately coupled with Simula. The notion of an object was adopted by others, and plays the central theme in the language Smalltalk [4]. However, the paradigm of simulation and quasi-concurrency was dropped - or at least fell into the background.

An OOP Language?

First, it is worth noting that applications which fit the object-oriented view usually involve a large number of data elements, most of which have a transient existence. The primary requirement to produce such applications is the availability of *dynamic data structures,* often expressed as records accessed by pointers. The necessary mechanisms are dynamic data allocation and (preferably automatic) retrieval.

We can also identify two essential requirements for an object-oriented language:

- It must be possible to define templates (of objects) consisting of variables *and* procedures. Templates were given the name *class,* and instances of class are said to be *objects.* Procedures defined for a class are called *methods,* and invoking a method is called *sending a message.*

- It must be possible to derive new classes from existing classes. A derived class is *related* to its base class (from which it is derived) by the fact that it adopts the latter's variables, adopts/replaces its procedures, and may add new variables and procedures. A derived class is compatible with the deriving class, in the sense that an instance of the derived class can be substituted for any object of the deriving class.

These compatibility rules imply that a procedure of an object may be invoked without reference to its exact identity, because the object may be an instance of many of the derived classes. This explains the use of the term *sending a message* instead of *calling a procedure.* The meaning of the message is known, the interpreting procedure is not. The dispatch of a message can now occur in a place where the actual procedure is unknown, in particular in a module that lies below the module in which the procedure is defined. Hence, such calls also are known under the term *upcall.* Figure 1 shows a comparison of conventional and object-oriented terminology.

We are now in a position to investigate the suitability of Modula [3] for object-oriented programming. The need for dynamic data structures is satisfied (although most implementations do not contain automatic storage retrieval). The first requirement is also satisfied, through the existence of procedure types. Objects can be represented as records, their methods as procedure-typed fields. Sending a message turns out to be an indirect call to a procedure via a procedure-typed variable of this sort.

However, the second requirement is not met. It is impossible to derive a type T1 from

```
TYPE  Point2 = RECORD x, y: INTEGER END
      Point3 = RECORD (Point 2) z: INTEGER END
VAR   p2: Point2; p3: Point3;


p2 := p3   corresponds to   p2.x := p3.x; p2.y := p3.y
```

Figure 2 - Deriving 3D type from 2D type

IN C.A.S.E. YOU'RE INTERESTED IN QUALITY!

See us on stand 203
at Software Tools '90
Wembley 12-14th June 1990

C.A.S.E. – Computer Aided Software Engineering.

VERILOG is setting new standards in real-time software development, with a range of CASE tools that addresses each stage of the development lifecycle, from specification through to testing and validation.

AGE is a powerful, integrated workbench for requirements analysis, specification and real-time system design. Built-in simulation and verification features ensure the quality and integrity of the system.

LOGISCOPE is the only software testing tool which provides complexity and test coverage results for a full range of development languages. Improved code quality and test effectiveness reduces the cost of software development, testing and maintenance.

VERILOG's software development tools provide powerful features to enhance quality and improve productivity, transforming what was an unstructured and insular process into a formalised industrialised procedure.

VERILOG's tools are already in use on numerous major projects in the UK and Europe. For more details on our complete product range call William Rigg on **081-940 2212.**

CIRCLE NO. 886

VERILOG

a type T0 such that T0 remains compatible with T1 (apart from the trivial cases of identity and subranges).

We could leave the topic at this point. However, it is worth investigating whether an object-oriented style might be used, if certain safety properties of Modula are sacrificed. As was pointed out in [9], a possible approach lies in using the special features contained in the module SYSTEM; in particular, the ADDRESS type offers a solution. *(Modula's* ADDRESS *type is a generic pointer type, similar to C's* void * *type - Ed.)* In anticipation of the need derived types, the declaration of the basic type is provided with an additional field, say ext, of type ADDRESS. It is then possible to produce derived objects, by assigning to ext a pointer to a record type containing the extra variables and procedures.

Leaving aside the question of the additional indirection in accessing the new fields, the crucial drawback to this approach is that the compiler's type checking capability has been crippled. A program using this recipe is potentially as unsafe as assembler code. Type safety should be the last property of a high-level language that we are willing to sacrifice. The low-level facilities in Modula-2 were provided with the intent that they be used sparingly, in cases where access to special machine resources are needed, and that they be isolated in small driver modules. I caution against their instalment as central instruments to be used throughout entire programs.

What we must do, from the point of view of language design, is find a solution which is neither a fix nor a trick, but which properly integrates the new requirement with the existing properties, and fully complies with the concept of type validation through textual inspection (viz compile-time checking).

Extending Modula-2

Our approach to adapting Modula-2 for object-oriented programming lies in extending the language with features that fit into the existing framework, allowing a precise and concise definition based on well-understood mathematical concepts. If we concentrate on the fundamental requirements, only a single new facility is actually called for: the one for introducing derived classes of objects.

Fads appear in times of acute difficulties, and are praised as medicine against the major ills

In accordance with the basic principle of introducing as few concepts as possible, and recognising the strong similarity between types and classes, we equate the term 'object' with 'instance' (of a record type) and 'class' with 'type'. A derived class (subclass) then corresponds to an *extension* of a record type [5]. To take a geometric analogy, if a record type is defined to represent a point in n-dimensional space, with each dimension represented by a field of the record, an extension - adding extra fields - increases the dimensionality of the space. An assignment of an instance of a derived class (ie a variable of an extended type) to an instance of its superclass (ie to a variable of the base type) corresponds to a *projection* of the variable's value onto the subspace spanned by the base type. Figure 2 gives an example of this, using two-dimensional space for the base type (Point2) and three-dimensional space for the derived type (Point3) .

The concept of extension is also applied to pointer types, so it becomes possible to construct heterogeneous structures, whose relating pointer type is bound to a node type R. The nodes of the structure may then be of different extensions of R, say R1, R2 and R3. Clearly, a need for determining the actual (extended) type of a node referenced by a pointer (bound to R) arises. Oberon, a Modula-2 based object-oriented language, provides it in the form of a *type test*, implemented as a Boolean factor with the operator IS.

```
CLASS Viewers =                          TYPE Viewers =
     BEGIN x, y, w, h: INTEGER;          RECORD c, y, w, h: INTEGER;
      METHOD restore (T: Text)           restore:PROCEDURE (T:Text)
         BEGIN ...                       END
      END restore
     END


v := Viewers.New (X, Y, W, H)            NEW(v);
                                         v.x := X; v.y := Y; v.w := W; v.h := H;
                                         v.restore := Restore
```

Figure 3 - Class-centred versus instance-centred syntax

We emphasise that the principal achievement of Oberon [6,7] is that the concepts of types and classes were *united*. The coexistence of two distinct notions, representing virtually the same concept, was avoided.

Syntax

Some more should be said regarding the representation of methods through procedure-typed record fields. In object-oriented languages, a class declaration looks just like a record declaration, with procedure declarations (or at least procedure headings) tagged on. This approach has consequences and some advantages. In Modula and Oberon, the corresponding procedure-typed field assumes the role of a variable. Hence the actual procedure must be assigned to it explicitly, each time an instance of the record is generated. This can be regarded either as a burden (and a source of mistakes) or as an additional degree of freedom (and power). Yet most typical applications bind the *same* procedure (handler) to all instances of a class: the view of methods is *class-centred*. Oberon's view is *instance-centred*. Figure 3 illustrates this contrast.

In an implementation of the class-centred view, each instance will contain a hidden pointer to the same table of procedure references. In an Oberon implementation, each instance contains direct (and duplicated) references to the installed procedures. This is clearly undesirable, if there are many of them.

Another advantage of the class-centred view, and of declaring procedure bodies within the class declaration, is the possibility of referring directly to the object fields (x, y, w and h in Figure 3) from within the procedure. This leads to a convenient formulation of sending the message restore to the object v:

```
v.restore(T)
```

This form is entirely consistent with the notation for field designators such as v.x. In the above call, v plays a double role as *distinguished parameter*. It qualifies the method name (via the class of v), and it represents a parameter of the call, namely the variable v. In Oberon, this abbreviating form is not possible, and the two roles are clearly disjoint:

```
v.restore(v, T)
```

The distinction between class- and instance-centred view with regard to method definitions can be viewed in another way. In the former case, methods are declared as procedure-typed *constants*, in the latter as procedure-typed variables. The restrictiveness of the class-centred approach becomes apparent when one considers subclasses (type extensions). Usually, a subclass contains methods different from those of its superclass. As they are declared as constants, a new language 'feature' is required: the ability to *override* the definition of the superclass. In class-centred implementations, overriding is achieved by the provision of a distinct method table for each subclass. In the instance-centred view, adopted by Oberon, no such additional mechanism is necessary. Nor has the additional notion of overriding any place, as it occurs through a regular, explicit assignment.

A class-centred derivative of Oberon has recently been devised and implemented [10]. It showed that the additional complexity of the compiler remains within tolerable bounds. The unanswered question is rather whether the notational conveniences justify the *conceptual* complications.

Object-oriented languages typically confine objects to be dynamically allocated records, which are referenced via pointers. In Oberon, such a restriction would have to be defined through an explicit, exceptional rule. As it stands, the language permits both dynamic *and* static variables of a record type; these variables are, therefore, extensible. In practice, the type extension concept turned out to be extremely useful in the case of static variables passed as reference parameters to procedures. If only dynamically-allocated records were permitted, the consequence would be the use of dynamic allocation when, conceptually, a variable should be declared as static and local, because of its transitory nature. This, in turn, may have grave consequences on the efficiency of an implementation. Oberon's generality turns out to be a significant benefit. Treating every variable as an object is a mistake.

Conclusions

Apart from convenient syntactic constructs, an object-oriented language features declarations of procedures bound to data structures (records), and offers the possibility to declare structures (extensions) that are derived from other structures and are type-compatible with them.

In Modula-2, it is possible to adopt the object-oriented paradigm, by resorting to low-level facilities and sacrificing the most important asset of a high-level language: the guarantee of type consistency.

Almost the entire operating system was better programmed without OOP

The language Oberon extends Modula-2 with the necessary facility: *type extensions*. There exist, however, some differences between Oberon's object-oriented facilities and those of other object-oriented languages. The most important of these is that in conventional object-oriented languages, procedures/methods appear as constants in the declaration of the record type/class. In Oberon, they appear as variables (record fields). In the first case, methods are guaranteed to be the same for all instances of a class. Oberon's methods may differ from instance to instance, and need to be explicitly installed whenever an instance is generated. With the typical object-oriented languages, the redefinition of methods (installed as constants) requires the additional concept of overriding. No such facility is needed in Oberon.

As a result, Oberon is conceptually simpler, and Oberon implementations are not burdened with additional class mechanisms. On the other hand, other object-oriented languages may offer somewhat more convenient notational facilities, and provide extra security by guaranteeing the constancy of declared methods for all instances of a class. This results in improved efficiency of upcalls. We consider this as a negligible advantage, since we believe that the object-oriented paradigm should be employed very selectively. In the design of an entire operating system [8], we found that almost the whole system was advantageously programmed in the conventional style. The object-oriented style was restricted to the viewer system, which provides distributed control. It is wise to use upcalls sparingly.

A much more significant contribution to efficiency is the generalisation of the type extension (subclass) concept to static variables, and in particular in their use as procedure parameters.

The most significant aspect of Oberon is that it supports both object-oriented programming and the conventional style, and that it guarantees full type-consistency checking. Oberon differs from other languages, because of an underlying conviction about language design: one should strive for simplification, through integration of similar concepts, rather than for complication through the addition of new facilities, similar to those which already exist.

This article was first presented as a paper at the First International Conference on Modula-2, Bled, Yugoslavia, Oct. 11-13, 1989. Many thanks to the Josef Stefan Institute, Yugoslavia, and to Professor Wirth himself, for permission to reprint it. Complete copies of the conference proceedings, priced £20, are available from: Modula-2 (E-4), J Stefan Institute, Jamova 39, 61111 Ljubljana, Yugoslavia. Phone 010 38 61 214 399.

EXE

References - In chronological order.

1. *O.-J. Dahl, K Nygaard. Simula: An Algol-Based Simulation Language. Comm. ACM 9,9, (Sept. 1966), 671-678.*
2. *G Birtwistle, et. al. Simula Begin. Auerbach 1973.*
3. *N. Wirth. Programming in Modula-2. Springer-Verlag, 1982.*
4. *A. Goldberg, D. Robson. Smalltalk-80: The Language and its Implementation. Addison-Wesley 1983.*
5. *N. Wirth. Type Extensions. ACM Trans. Prog. Lang. and Systems, 10, 2 (April 1988), 204-214.*
6. *--. From Modula-2 to Oberon. Software, Practice and Experience 18, 7, (July 1988), 661-670.*
7. *The Programming Language Oberon. Software, Practice and Experience 18, 7, (July 1988), 671-690.*
8. *J Gutknecht, N. Wirth. The Oberon System. Software, Practice and Ex*perience, 19, (1989).
9. *G. Blaschek, Implementation of Objects in Modula-2. Structured Programming, 1, 3, (June 1989), 147-156.*
10. *H. Mössenböck, J. Templ, R. Griesemer. Object-Oberon, An Object-Oriented Extension of Oberon. Report 109, Dept. Informatik, ETH Zürich (June 1989).*

History of a Useful Illusion

Who said history could never be objective? Richard Drake tells the tale of a technology born in obscurity in the 1960s, and acclaimed as the great hope for the 1990s.

It could only have been created by a man who knew what it felt like to be à CPU. Shortly after the war, the Norwegian government, which was struggling to set up a nuclear power industry, needed a digital computer to perform some complex and repetitive computations. Of course, there was no cash for such a machine, so instead the Norwegians filled a room with soldiers, including one Private Kristen Nygaard (sounds like 'knee-guard'), and kept them there for several weeks, simulating the calculations of the inaffordable machine. This experience must have rankled, because many years later Kristen Nygaard, with his colleague, one Ole Dahl, turned the tables on the electronic CPU by designing two successive computer languages that allowed effective simulations to be programmed on it.

The first of these was Simula I, an extension of Algol 60, which introduced the concept of an 'activity' or process description. By the middle of the 1960s, Nygaard and Dahl felt they had identified a number of key points through their work with this language. In particular:

- The distinction between a piece of program text and an execution, or a 'dynamic instance' of it.
- Data and operations belong together, and most useful program constructions contain both.
- Processes often share a number of common properties, both data and actions, whose declarations should not have to be repeated.
- A garbage collection scheme is required for removing dynamically created processes that are no longer needed.

The last idea was borrowed in part from LISP world - the first of a number of ways in which LISP was to influence the direction of OOP. These concepts paved the way for an event that is widely recognised as the birth of object-oriented programming.

Born in Obscurity

In 1967, Nygaard and Dahl created Simula 67 (actually the final definition was, like most software babies, delivered late, in May 1968). Simula 67 was a general purpose programming language, later to become famous as the first object-oriented programming language.

If this was the birth of OOP, its conception can probably be traced to a wintry Norwegian day in December 1966, when the solution to the problem of repeated definitions of 'common properties' was finally drafted. By this time, Nygaard and Dahl had renamed activities 'classes', and had begun to ponder how common properties might be grouped together in a class from which other 'subclasses' could be derived. After careful thought, they arrived at an open syntax for subclasses (contrasting with Pascal's closed syntax for variant records) that would allow the extension of an existing class, without any need for modification of the original source code.

The problem used to illustrate subclassing was the simulation of a toll booth on a bridge with a queue of vehicles that may be either cars or trucks. A sketch of the solution, using the class `Link` as the basis for a linked list of vehicles, is shown in Figure 1. The advantage of this approach is that if, later on, the program needs to be extended to cope with buses, none of the existing code for `Vehicle` need be changed. Thus it became natural to construct hierarchies of classes in Simula 67, a key feature of object-oriented programming to this day.

```
class Link; ...definition of data and actions... ;
Link class Vehicle; ... ; (Vehicle defined as subclass of Link)
Vehicle class Car; ... ; (Car defined as subclass of Vehicle)
Vehicle class Truck; ... ; (Truck defined as subclass of Vehicle)


Later on, the program can be extended to cope with buses, without
altering the code for Vehicle:


Vehicle class Bus; ...data and actions specific to Bus here...;
```

Figure 1 - Vehicle subclassing example

Subclassing led to further language changes, including relaxing compatibility rules so that, for example, references to instances of Car or Truck could be assigned to variables declared as Vehicle, allowing general code to handle a queue of vehicles without needing to know all the possible subclasses. In addition, where an action was required that was defined in more than one class, there was a need to resolve which version(s) should be used. A construct was invented to combine procedures defined in a class and its subclass, and the principle of dynamic or run-time binding of procedures was established, to let general code specify the appropriate action (such as travel at average speed) without necessarily know-

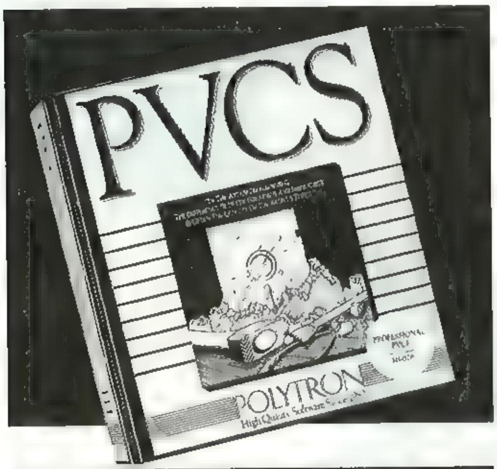
POLYTRON

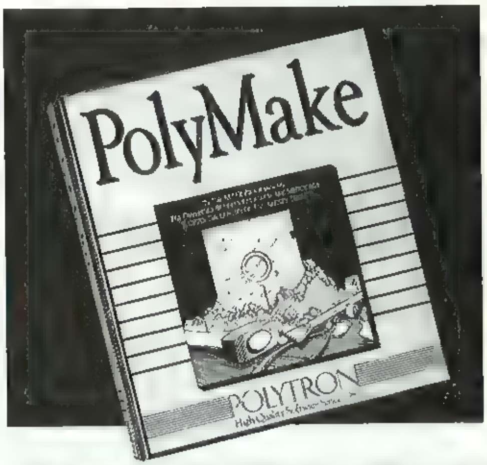
SOFTWARE DEVELOPMENT TOOLS FROM

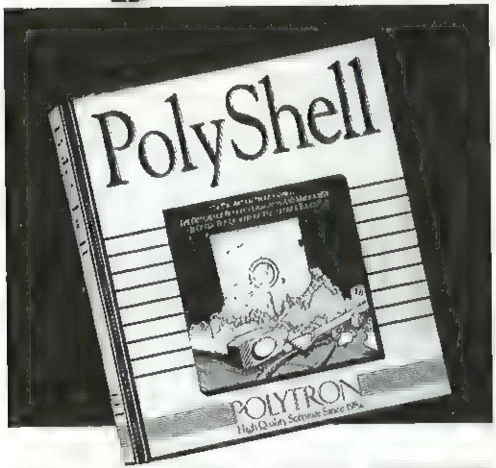
SOFTWARE GENERATION

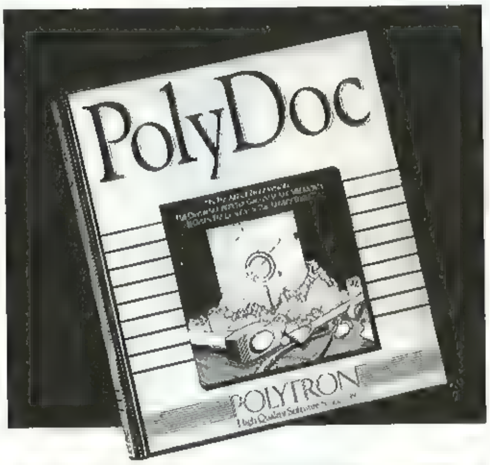
SOLE DISTRIBUTOR

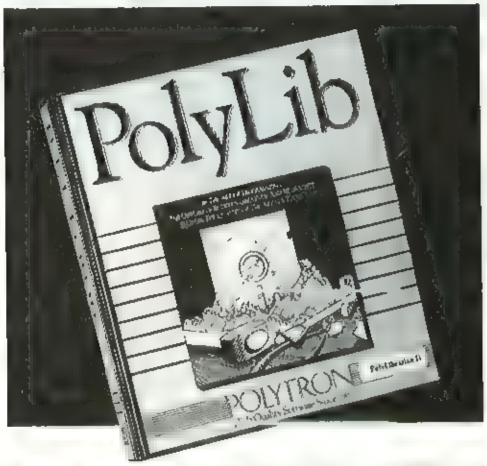
0462 422525

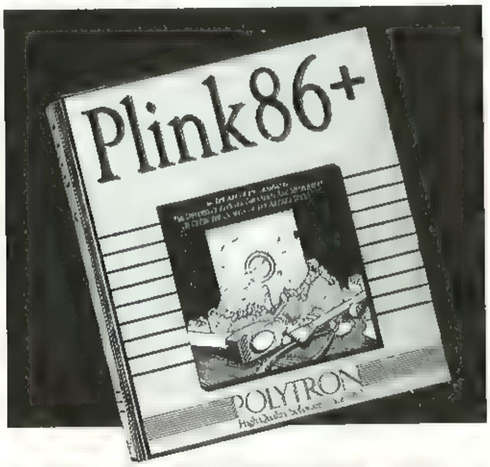










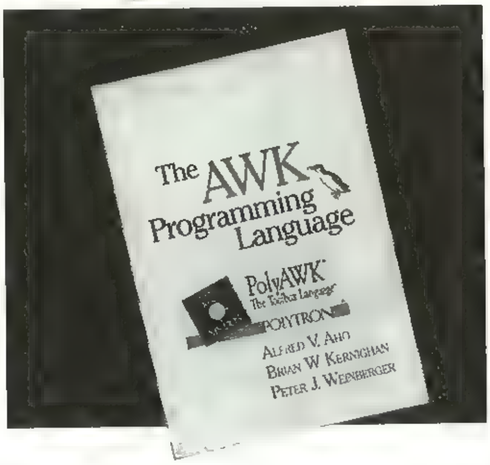


APPROVED DEALERS

Roundhill Computer Systems Limited
Orchard House
Ogbourne St. George, Marlborough
Wiltshire SN8 1SU
Tel: (0672) 84535

Computer Solutions Limited
Canada Road, Byfleet
Surrey KT14 7HQ
Tel: (09323) 52744

Grey Matter Limited
Prigg Meadow, Ashburton
Devon TQ13 7DF
Tel: (0364) 53499



APPROVED DEALERS

Frontline Distribution Limited
Intec 1, Wade Road, Basingstoke
Hampshire RG24 0NE
Tel: (0256) 463344

Reflex Technology Limited
9 Buckingham Place
Bellfield Road, High Wycombe
Bucks HP13 5HW
Tel: (0494) 465418

System Science
3-5 Cynthia Street
London N1 9JF
Tel: 01-837 6411

CIRCLE NO. 888

SINCE WHEN HAS AN APPLICATION DEVELOPMENT TOOL BEEN A NECESSITY? SINCE TODAY!

If you write applications in any of the leading dBASE languages, here's a great way to develop really professional programs in record time and with the minimum of effort: the Lamaura Data Dictionary.

It gives your PC development system facilities normally found only on minis and mainframes.

Now you can try out data structures and file relationships at the right time...... **BEFORE THE APPLICATION IS WRITTEN.**

You can store information about applications on the Data Dictionary Notepads. Then, all the information about fields, records, Tables (.DBF files), and applications is centralised for easy reference.

It's also handy for understanding the relationship between fields in a Table and indexes on applications already written by someone else.

With picture clauses and data validation on input fields, referential integrity and circular reference checking, you'll easily realise why the Lamaura Data Dictionary is such a necessary development tool.

There are versions for single, 5, 25 and 99 users. The three multi-user versions include message passing with real time information updates, and three level security from project supervisor down. Naturally, our documentation is thorough and concise, even including a section on data base design.

Take a little time to clip the coupon now, and we'll send you an information pack that demonstrates today's most necessary application development tool.

TECHNICAL INFORMATION

Reads and creates .DBF file formats

Supports dBASE III Plus (.NDX), Clipper (.NTX) and FoxBASE + (.IDX) index file formats

Requires 450K bytes RAM and DOS 3.0 and above.

Lamaura and the Lamaura logo are trademarks of Lamaura Development Limited. dBASE is a registered trademark of Ashton-Tate Corporation. dBASE III Plus is a trademark of Ashton-Tate Corporation. Clipper is a registered trademark of Nantucket Corporation. FoxBASE + is a trademark of Fox Software Inc. Copyright © 1989 Lamaura Development Ltd. All rights reserved.

Distributed by
Nantucket UK Ltd through Softsel, Frontline and Software Ltd Tel 0707 373600 for details.

Lamaura™

Lamaura Development Ltd
Rabley Park Ridge Potters Bar Herts EN6 3LX
Telephone (0707) 43278

Please send me an information pack about the Lamaura Data Dictionary.

Name: ______

Title: ______

Company: ______

Address: ______

______

Tel: ______

.EXE 1

Return to: Nantucket UK Limited,
8 Bridgegate Centre, Martinfield,
Welwyn Garden City, Herts AL7 1JG.
Telephone: 0707 373600

THE BEST THING NEXT TO YOUR dBASE OR CLIPPER OR FOXBASE OR ...

CIRCLE NO. 889

ing exactly which class(es) would be involved at run-time.

Like any new baby, object-oriented programming needed a name. In Spring 1967 the term 'object' was first coined, as a more neutral term than 'process' for the instance of a class. Little did the creators guess that such a trivial change would have such a dramatic effect on software marketing over twenty years later, as desperate attempts to draft and trademark the latest fOOlish anachronism or OBJECTionable product name fill the waking hours of copywriters and lawyers, all to sell software products that are frequently inferior to Simula 67.

How has OOP gone from obscurity (ie not being invented in America or mass-produced in Japan) to become the software superstar of the 1990s? Well, if you can't be born in the USA, the next best thing is to be adopted there. This is what happened to OOP in an all-American research lab in the 1970s...

A Secluded Childhood

Around 1970, a young jazz musician, and sometime student at the University of Utah, was introduced both to Simula 67 and Ivan Sutherland's innovative 'Sketchpad' program, which used both a bitmap display and a pointing device. He argued that a combination of the inherently powerful concepts in Simula and the user interface of Sketchpad should be used to create a 'reactive engine', a software language and environment that would be far easier to learn and use than other development systems, and could form the basis for a hand-held personal computer called the 'dynabook'.

Alan Kay didn't get everything right about the future of personal computers (he just missed some minor points, like the fact that a computer as primitive as the Apple II could attract a reasonable market by 1980...). But in the days of mainframes, batch programs and punched-card input, his vision was recognised and backed by the newly-formed Xerox Palo Alto Research Centre (PARC), where he set up the Learning Research Group, with the aim of creating a software system easy enough to be used and 'programmed' by children.

Kay's vision included a language, Smalltalk, which would be uniformly object-oriented - that is, all program entities, including numbers and strings, would be instances of classes, ie objects. In order to emphasise the uniform computational model of the system, the term 'sending a message' was used to describe the calling of action belonging to an object. This became the only way anything in the system got done. For example, the syntax '3 + 4' was shorthand for sending the message '+' to object '3' with parameter '4'. The code within a class which implemented the message was called a 'method'.

Kay carried out studies of the way children learned to use the system. The feedback from these experiments influenced the way that the user interface of Smalltalk developed. Kay used the term 'user illusion' to describe how objects should be represented on the screen, and originated the metaphor of overlapping windows and pop-up menus. Larry Tesler (who joined Apple in 1981 to help create the Lisa/Macintosh) introduced the complementary ideas of modeless operation, object/verb editing and powerful online browsing facilities for existing source code. Dan Ingalls added a debugger which, even today, shames most of the competition. Other members of the team, which included leading computer scientists like Adele Goldberg and Peter Deutsch, implemented integrated text editing and fast incremental compilation. By 1980, they had created what was the most powerful programming language environment of its time - a position Smalltalk arguably retains today.

It could only have been created by a man who knew what it felt like to be a CPU

In keeping with the conceptual simplicity and the interactive nature of Smalltalk development, the language does not require the class or type of variables to be declared, as in Simula. All objects, except those belonging to 'primitive' classes such as `Boolean` or `SmallInteger`, are created dynamically as required. These are accessed through double-indirected pointers, and garbage-collected by the environment when they are no longer referenced. The programmer has no worries over memory management... but, there again, he has no control over it either, as inveterate C programmers lament. Rather confusingly to Smalltalk novices, classes themselves are objects; instances are created by sending the message `new` to the class in question. Subclasses automatically inherit data fields ('instance variables') and methods from their 'superclasses'. The environment supports various incremental changes to classes, including adding or removing instance variables, even when the classes have existing instances.

Since 1978, most Smalltalk implementations have used a virtual machine model, to maximise portability. The system is similar to the Pascal p-code approach. Source code is compiled into intermediate 'byte-codes', and these are interpreted at run-time. In the last few years, there have appeared implementations which generate native machine code. Consequently the performance of Smalltalk object code, although inferior to that produced by optimising C compilers, has improved steadily. Run-time errors are painless. A new window, called a notifier, appears to report the error - the most common being: sending a message to an object that doesn't 'understand' it. The debugger can be activated directly from the notifier. Very often the source code can be debugged, and the process resumed, without losing results of the execution so far.

Socialising

In the last decade, the happy picture of steady development, leading to the uniformity and productivity of Smalltalk, is spoilt. OOP experimented with new things, mixed with all the wrong people and, like most teenagers, got in a real mess. But it also learnt a lot; in particular it learnt to relate to real-world software developers, and how they build (and fail to build) commercial systems. Slowly OOP began to influence fundamentally the way systems are put together, a trend that looks set to continue in the immediate future.

Even in the 1970s, object-oriented dialects of LISP had begun to emerge. Flavors and LOOPS are the best known products from this period. Subsequently, adding objects to any and all languages became high fashion, with more or less pleasing results. In my view, 'adding objects' has no real validity unless the resulting language supports at least three things:

- Encapsulation (data and functions/procedures/methods defined together) to create classes (object types in Object Pascal, structs or classes in C++).
- Inheritance of data and functions/methods from super (or base) classes by sub (or derived) classes (C++ terms in brackets, forget Object Pascal!).
- Dynamic or run-time binding, where the result of sending a message/invoking a

Figure 2 - CLOS Multi-Methods

```
;Generic function install
(defgeneric install (software-product operating-system)
  (:documentation "Installs software on the operating system."))


;Method 1 - when both arguments are valid
(defmethod install ((sw basic-product) (os basic-os))
  ;body of method goes here
  )


;Method 2 - if install is called with an invalid OS argument
(defmethod install ((sw basic-product) non-os)
  (error "Cannot install because ~A is not
         a recognised operating system." non-os))


;Method 3 - if install is called with an invalid product argument
(defmethod install (non-product (os basic-os))
  (error "Cannot install because ~A is not a
         recognised software product." non-product))


;Method 4 - both arguments invalid
(defmethod install (non-product non-os)
  (error "Cannot install because ~A is not a
         recognised software product and ~A is not
         a recognised operating system."
         non-product non-os))
```

function depends on the class of the object involved at run time (automatic in Smalltalk and most OOP languages - specified by virtual functions in C++).

Did anybody notice an exponential increase in the complexity of the jargon? This was the first (inevitable) result of letting the unwashed masses get hold of a neat, uniform idea like OOP in Smalltalk.

Objective C, from Stepstone Inc, was the first on the C-with-objects bandwagon. It added a Smalltalk-like message sending syntax to C, but let the language do its own thing with integers and characters. The mismatch between 'objects' and these more elementary data types proved aggravating, but not fatal. With a useful library of basic classes, Objective C was later on adopted as the main development language for Steve Jobs' NextStep environment on the NeXT cube.

Meanwhile, Bjarne Stroustrup extended C into C++. Because he was careful to stay as close as possible to C, and because he worked for AT&T Bell Labs (a slightly unfair advantage, some might say) C++ looks set to become the standard for OOP, just as C became the standard for conventional programming. Unfortunately, C++ is, at present, several times worse than Smalltalk for creating *really* general purpose class libraries, although it is more memory and processor efficient. It has no automatic garbage collection, no way of creating general purpose collection classes (because of its strong typing and lack of parameterised types), objects have no knowledge of their class at run-time and classes are not themselves objects.

Even with an environment like Objectworks for C++, or a cheap and cheerful one, such as supplied by Zortech on the PC, the lack of a rich class library for C++ (caused partly by deficiencies in the language) means at present that C++ programmers cannot get a feel for the full potential of OOP. Parameterised types should make a difference here, and are promised by Stroustrup some time soon...

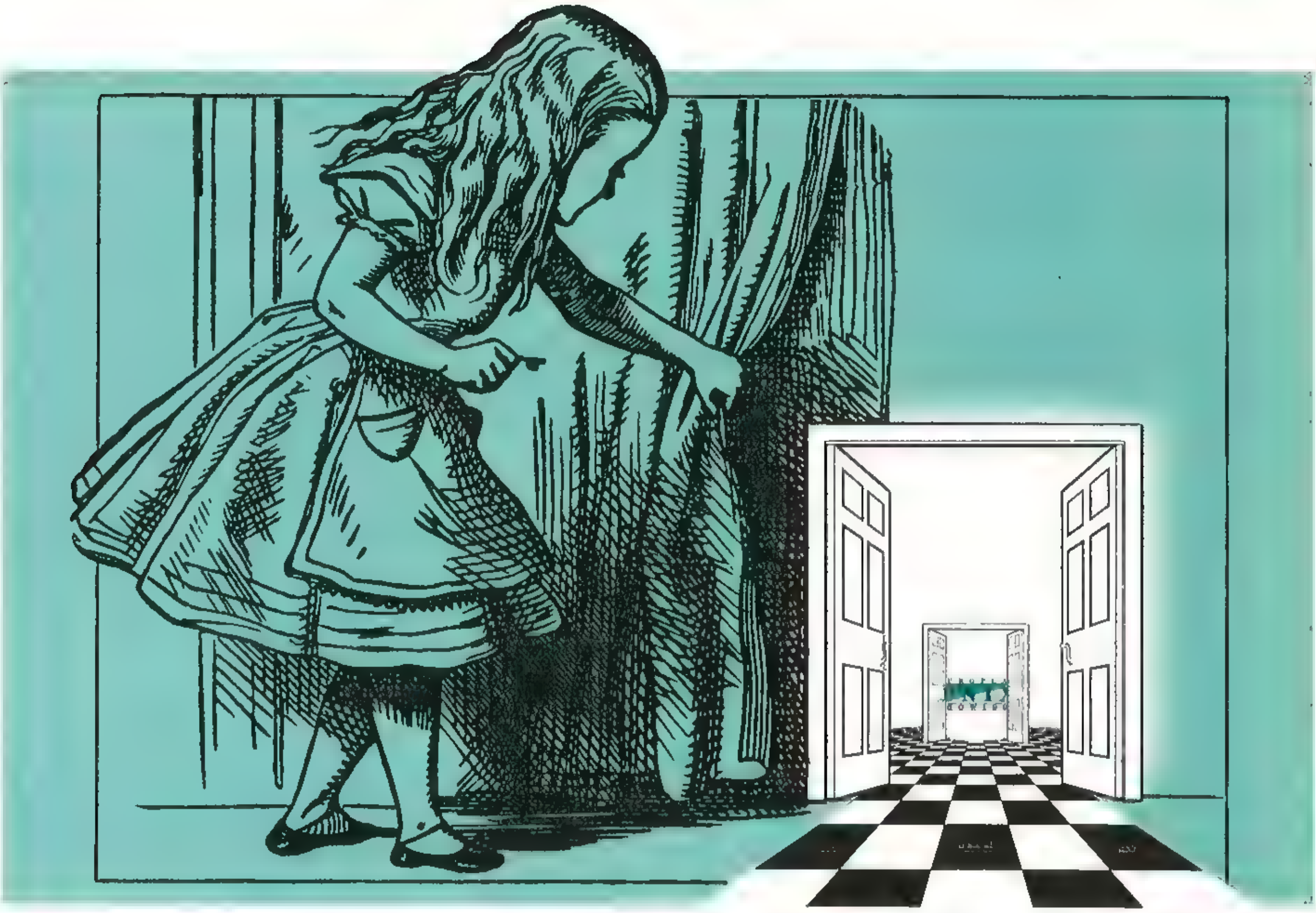
In the same vein as AT&T, Apple decided to add objects to Pascal (with a little help from Niklaus Wirth, the inventor of the language) to create Object Pascal. Recently Microsoft and Borland claimed to have released versions of the same language. (Closer analysis by Paul Smith, in .EXE Magazine Sept-Oct '89, suggested that Microsoft implemented the correct language badly, and Borland invented its own language, and implemented it quite well.) Apple, at least, under the guiding hand of Tesler (and now Kay and Ingalls too), provide a decent set of classes. MacApp (as the Macintosh class library is known) supports the Macintosh user interface, and thus does give novices a reasonable amount of OOP for their money. Object Pascal suffers from some of the same problems as C extensions; there is an uneasy cohabitation of Pascal's data types alongside the new object types.

As far as pure OOP languages are concerned, there are three of reasonable commercial significance. Smalltalk exists in two dialects - 80 and V - which are identical, more or less, in their syntax, but differ in their class libraries. Objectworks for Smalltalk-80 is the original system, produced and 'ruggedised' by ParcPlace. This runs on almost any brand of workstation, not to mention the Macintosh and 386 class PCs. It remains the most complete OOP environment but suffers at present from supporting only its own 'old-fashioned' windowing system (it was the first, after all). On the other hand, its portability means that applications written in it can be ported between platforms in a matter of minutes. I can testify the truth of this from first hand experience, having last year moved a large system back and forth between Macintoshes and Apollos for a UK clearing bank.

Smalltalk/V, from Digitalk, runs under MS-DOS, OS/2 PM and the Mac, and supports the native windowing in the last two environments. Although it does not have all the richness of 80's development environment, it has enough to give a good flavour of life in the OOP fast lane at a competitive price. It is also the proud owner of a native code compiler for OS/2 PM.

The two other OOPLs of note are Eiffel and CLOS, the Common LISP Object System. Eiffel is Bertrand Meyer's elegant language, designed with modern techniques of software engineering in mind. It includes class assertions, post and pre-conditions and clean error-handling for messages. Both Eiffel and CLOS (and, incidentally, V2.0 of C++) support multiple inheritance, in which a subclass can inherit from more than one parent class. Proponents of Eiffel, in particular, claim that this facility represents a major improvement in the expressive power of an OOP language.

CLOS also introduces a more general and powerful mechanism for selecting the method executed at run-time for a given message, called multi-methods. A CLOS generic function (message) with more than one parameter invokes a method based on the classes of all the parameters, not just the first (the first traditionally being the object to which the message is 'sent' in other OOPLs). So the generic function `install`, for example, which installs a new software product on an operating system, will invoke a method depending on the class both of the first parameter (the product) and the second (the operating system). See Figure 2 for a simple application of this; the example is taken from Sonya



Open your doors to open systems

Explore the endless computing possibilities at the European Unix Show 1990.

Draw back the curtains on the rapidly developing open systems technologies and discover for yourself the products and services best suited for your continued business success.

Over 150 leading companies are waiting to provide you with tailor-made solutions for your every business need.

'Open Solutions '90' Conference

Take a look into your future at the three-day 'Open Solutions '90' Conference.

Benefit from the opinions of international experts and compare your business requirements with those of leading open systems users at separate managerial and technical sessions.

Free EMATCH Visit Planning Service

Ask for your Free EMATCH Visit Planner – we'll give you a personalised programme pinpointing the products and services most relevant to your business needs – so you'll get the very best from the European Unix Show 1990.

Call us today on the Ticket Hotline, or complete the coupon for your complimentary tickets, conference details and Free EMATCH Visit Planner.

Remember to put 19-21 June, Olympia in your diary, so you don't miss your only opportunity to see the latest open systems developments at the largest event of its kind in Europe.

EUROPEAN UNIX SHOW
The Open Door to Open Systems.

TICKET HOTLINE 01-404 4844 x 2620 (071-404 4844 x 2620 from 6th May)

E·U·R·O·P·E·A·N
UNIX
S·H·O·W·1·9·9·0
NATIONAL HALL OLYMPIA 19-21 JUNE

EXE 5/90

Please send me complimentary tickets to the European Unix Show together with details of the 'Open Solutions '90' Conference and Free EMATCH Visit Planner.

Name____________________

Position____________________

Company____________________

Address____________________

____

Postcode__________ Telephone__________

To European Unix Show, EMAP International Exhibitions Ltd., 12 Bedford Row, London WC1R 4DU.

OPEN SYSTEMS

An official journal of the European Unix Show. Unix is a trademark of AT & T in the USA and other countries.

CIRCLE NO. 890

Keene's excellent book *Object-Oriented Programming in Common Lisp* (Addison-Wesley, 1989).

Because CLOS seamlessly integrates dynamically-bound generic functions with standard LISP functions, I class it as a 'pure' OOP language, despite its pre-OOP origins.

As well as sincere attempts to graft OOP facilities onto conventional languages (Objective FORTRAN is now available for the NeXT cube!), less scrupulous, or just plain unintelligent, claims to be object-oriented have become boringly commonplace recently. Back in the 1980s, tired marketeers flogged the 'relational database' buzz-phrase to the limits of credulity until the man who invented it, Ted Codd, began publishing vast numbers of rules to define the term. The market-men then fell upon 'CASE' and 'OOP' with enthusiasm, realising that nobody seemed able to agree what either phrase described.

Object-orientation has suffered from a problem of definition since the 70s, because of the breadth of Alan Kay's vision for personal computing. Many definitions were based on Smalltalk-80, with the result that only those systems sharing selected user interface, design or language features with Smalltalk were object-oriented. Such an approach has dangers, but most experts agree that there are three essential mechanisms, as mentioned already, for any OOP language, which are common to languages described in this article:

1. Encapsulation (classes).

2. Inheritance (subclassing).

3. Polymorphism (dynamic binding).

OOP, so defined, is now 23, older and wiser because of the rough and tumble of the past decade. Everyone agrees it's high time it settled down and began to earn an honest living so that we can see what it really does have to offer...

Adulthood

Not surprisingly, given the amount of media and marketing copy given to OOP, it has its detractors and debunkers. How come something so good has been around for 23 years and *still* doesn't seem to have been used for any very major software systems? In any case, the whole thing is an illusion - you can get just the same effect in good old C, Modula-2 or Ada.

I would agree on the last point - in fact, the phrase 'user illusion' could legitimately refer to any software. All programs are abstractions; the poor old silicon has no knowledge of the the elaborately-named variables and procedures that the programmer writes, all it sees is a third hand translation which has been mauled by the compiler, linker etc. The question is: what is a useful illusion? The main claim for OOP is that it is a very useful metaphor for writing software systems, and there are languages that support the metaphor well.

Little did the coiners of the team 'object' guess what fOOlish abuses would follow

As for whether OOP has been used to build large systems, I have to admit: some, but not many. However, all the signs are that this is changing.

OOP, in maturing, has three needs: standards for languages and class libraries, the ability to 'live in' and work with existing hardware/software and, most significantly, a happy and stable marriage with the 'other side' of software technology, the database.

As Sam Goldwyn said, prediction is hard, especially when you're dealing with the future. But three trends are already noticeable and can be expected to increase in the 90s:

- *De facto* standardisation, as a few OOP languages are used a lot and the others are marginalised. C++ looks certain to figure as will, I dare suggest, Smalltalk and perhaps CLOS and Eiffel. Meanwhile, standard class libraries, such as those provided with Smalltalk, or NextStep, will become established, and will be usable from multiple languages (NeXT are pioneering here and the Object Management Group is also worth watching).
- Linked with this standardisation will be: support for all hardware platforms, 'callability' to and from conventional code and full support for look and feel standards such as PM, Mac, Motif and Open Look. Development environments will increasingly allow multi-language coding and linking, based on underlying object models.
- Closely impacting the above, object-oriented databases will gradually supplant relational databases, initially in fields such as CAD and GIS (Geographical Information Systems). These will increasingly support a single, uniform language for database definition, manipulation, querying and programming. Three main options identifiable today are: object extensions to SQL (eg Ingres 6.3), extended C++ (eg various US start-ups with large amounts of cash and sophisticated vapourware) and Smalltalk-like languages (eg GemStone from Servio Logic and Vision from Insyte, which is already in use analysing large quantities of financial data in Fortune 500 companies).

What's significant is that OOP languages already contain a lot of what is needed in a database language. Married with database concepts such as transactions and data security, they happily become one. The result is often called a 'persistent object system', because, from the programmer's point of view, little changes, except for the convenient fact that the objects you are dealing with persist after the program has ended, and so can be shared with other programs.

As persistent object systems, built from standard OOP languages such as C++ and Smalltalk, become widely available, OOP will have become a mature adult that will surely transform the software world. The next ten years will decide.

Postscript: sincere apologies to the 'parents' of OOP, Kristen Nygaard, Ole Dahl and their colleagues, for ignoring their significant work in applying Simula 67 and, more recently, in developing Beta, an innovative multi-paradigm language. I hope that the distinctive insights of the 'Scandinavian school' can be translated into globally successful OOP products in the 1990s, allowing the wider software community to benefit from the oldest and, arguably, the wisest body of experience available worldwide in building industrial-strength systems using objects. EXE

Richard Drake is Managing Director of Objective Computer Systems, a London-based consultancy specialising in object-oriented and database applications, including GIS and financial systems. He was one of the organisers of the third European Conference on Object Orientated Programming (ECOOP 89) last July.

The Tamer Transputer

David Gristwood looks at two ways of harnessing the power of a network transputers, using the C programming language.

To date, parallel processing has had very little impact on the IBM PC community. Yet the PC, with its open architecture, is the ideal host for plug-in parallel processor cards. Several such processors are available, one of the most popular being the Inmos transputer. In this article, I will explore some of the ways in which a network of transputers can offer increased processing power, using two different C compiler systems: 3L's Parallel C, and Helios.

To recap: the transputer is a RISC-like (Reduced Instruction Set Computer) processor with its own local memory and links for connecting to other transputers. The T800 series transputers have on-board floating point units, and are capable of an impressive 10 MIPS and 1.5 MFLOPS at 20 MHz. Each transputer has between 2 KB and 4 KB of internal RAM, mapped into the standard address space. The links, typically four per transputer, allow rapid inter-processor data transfer, and can operate at up to 20 Mbits/sec. Each link has its own controller, enabling the links to work in parallel with each other and the CPU.

The transputer's model of parallelism is based around the process. This is the basic building block for all systems. Programs are designed in terms of interconnecting sets of processes, which communicate along synchronised point-to-point channels. These channels are the 'outside' interface to the process, so that its internal design remains hidden.

```
#include "stdio.h"
#include "time.h"


#define  PACKETS     100
#define  ITERATIONS  1000


main()
{
  double      seed, x;
  long        u, v;


  seed = 1000.00;     /* starting seed */
  for (u = 0;  u < PACKETS;  u++)
  {
    x = seed / 2.0;   /* initial guess */
    for (v = 0;  v < ITERATIONS;  v++)
      x = x - ((x * x - seed) / (2.0 * x));
    seed += 1000.00;  /* next seed */
  }
}
```

Figure 1 - Sequential version of Root

The transputer provides a number of special operations to support this process model, eliminating the need for an operating system. Its instruction set includes commands to create and destroy processes. A micro-coded scheduler shares processor time between concurrent processes. Different processes on the same transputer use memory to synchronise communications, whereas processes executing on different units communicate via the transputer's links.

Sharing the Load

Enabling processors to work together is a fairly simple task. The same is not true for software components. Most transputer programming languages put the onus firmly on the programmer to code parallelism into his programs explicitly. In the case of C, which is primarily a sequential language, programs have to be broken down into a number of modules/tasks, each of which can be run in parallel. These tasks then communicate via channels.

The tasks and interconnecting channels that comprise a complete application must be configured for a processor network. Usually, the ideal network for a given program is one processor for each task, allowing maximum parallelism. In practice, a program must be able to run on a network of arbitrary size and topology. When there are fewer processors than tasks, some tasks must reside on the same processor and run concurrently. When there are more processors than tasks, it may be necessary to reroute messages.

The configuration of transputer programs generally takes place at either compile-time or load-time. The occam language allows compile-time configuration: tasks and channels can be assigned to particular processors and links. While this allows a program to exert maximum control of the network, the program must be re-coded and re-compiled for each new network. With load-time configuration, the technique adopted by the compilers in this article, tasks are created and compiled separately, and a configuration map is used to define where the tasks and channels are to reside. Programs can be developed independently of any network, and configured for different networks by altering the configuration map.

Root

To test the programming systems in this article, I wrote parallel versions of the a square root program, henceforth 'Root', which uses an iterative technique to calculate the square roots of a fixed sequence of numbers. The sequential version of the program is shown in Figure 1. The parallel versions use the network of transputers to reduce program execution time, by sharing the calculations out among the processors. Benchmark timings were taken for each system using a set of three weightings to give some indication of the communication overheads. The transputer benchmarks were run on MicroWay's Quadputer2 board, which consists of four T800s, running at 20 MHz. To provide some form of base measurement, the tests were also run using Microsoft C V5.0 on a 16 MHz 80386 PC, with no floating point processor. The results are shown in Figure 2.

Parallel C

3L, the supplier of Parallel C, is a company specialising in parallel versions of mainstream sequential programming languages. The company's philosophy is to provide an easy migration route for those wishing to develop applications on the transputer. Their compilers are designed to be used within the normal PC environment, looking

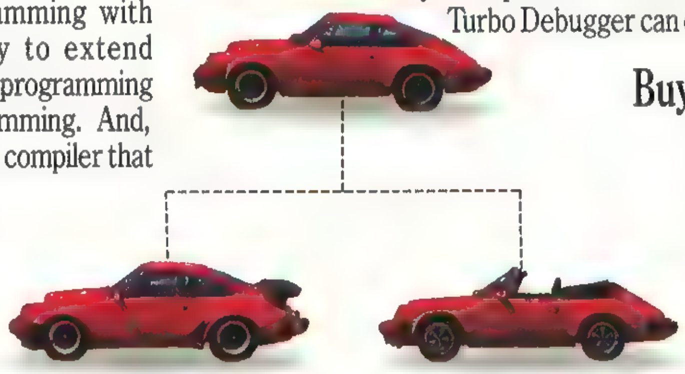
BORLAND INTRODUCES TURBO PASCAL 5.5 WITH OBJECTS

Be Objective.

Turbo Pascal,® the world-standard Pascal compiler, adds Object-oriented Programming with our new version 5.5. We combined the simplicity of Apple's Object Pascal language with the power and efficiency of C++ to create Turbo Pascal 5.5, the object-oriented programming language for the rest of us.

It's easy to extend yourself

If you're already programming with Turbo Pascal, it's easy to extend yourself from structured programming to object-oriented programming. And, Turbo Pascal 5.5 is the *only* compiler that is 100% source-code compatible with your existing Turbo Pascal 4.0 and 5.0 programs.



Inheritance provides powerful modeling capabilities by allowing objects to inherit attributes from other objects.

A fast object lesson

Object-oriented application programs more closely model the way you think. Objects contain both data and code. As in a spreadsheet cell, the value and the formula are together. Objects can *inherit* properties from other objects. For example, a Porsche Carrera inherits most attributes from base model 911, but it also sports a whale tail.

Turbo Pascal 5.5 Features

■ Inheritance ■ Static & dynamic objects ■ Constructors & Destructors ■ Object constants ■ Compiles @ > 34,000 lines/minute ■ New integrated environment tutorial ■ Hypertext Help with copy and paste ■ Enhanced smart linker & overlay manager ■ Support for 8087/80287/80387 ■ Integrated source-level debugging

Turbo Pascal 5.5's object-oriented extensions give you code that's easier to change, extend, and support.

Turbo Pascal 5.5 Professional with Turbo Debugger® and Turbo Assembler®

The award-winning Turbo Debugger now includes an object inspector and hierarchy browser. And Turbo Debugger can debug any size program.

Buy objectively

Turbo Pascal 5.5 costs £99.95 plus VAT and Turbo Pascal 5.5 Professional, including Turbo Debugger and Turbo Assembler costs £199.95 plus VAT.

For further information, simply talk to your dealer or call our PRODUCT INFOLINE for a copy of our Turbo Pascal 5.5 brochure on (0628) 771070.

BORLAND

Borland International (UK) Ltd.,
8 Pavilions, Ruscombe Business Park
Twyford, Berkshire RG10 9NN

CIRCLE NO. 892

Figure 2 - Benchmark timings

Root Benchmark Timings/seconds

		3L	Helios	Microsoft
		(T800)	(T800)	(80386)
PACKETS	ITERATIONS			
100	100000	18	170	6600
1000	10000	18	183	6600
10000	1000	19	238	6600

and feeling very much like their MS-DOS counterparts.

The latest addition to the 3L range is Tbug, an interactive source level debugger, which can be used with all the 3L compilers. It is similar to Microsoft's CodeView, but uses an overlapping (rather than tiled) windowing system, and boasts a very useful hypertext help system. Unfortunately, it doesn't use the same commands as CodeView.

The Parallel C compiler supports the K&R specification of the language, but includes several ANSI extensions, such as function prototyping. The manual which accompanies the compiler, although not a masterpiece of clarity, contains just about all the information needed.

Parallel C achieves its concurrency through a comprehensive set of library routines. As the compiler is designed to be similar to MS-DOS C compilers, sequential programs may be recompiled within the Parallel C environment with little or no changes. Obviously, such programs run sequentially on a single processor.

To exploit parallelism, a program must be written as a number of tasks, which communicate over channels using special I/O functions. Within a task, concurrent process threads may be created dynamically. These threads all share the same static, external and heap memory, but have their own private stack.

Threads may communicate using internal channels or shared memory. In the latter case, semaphores may be used to synchronise access to critical resources. Parallel versions of several common I/O routines, such as `printf()`, are provided to allow multiple thread access.

There are two basic configurations available under Parallel C. The first allows tasks and channels to be mapped explicitly onto a particular network, using a configuration file to define the hardware and software topology.

The second method is the the processor farm model. In this model, programs automatically configure themselves to run on any network of transputers. As shown in Figure 3, the processor farm model requires that the program is divided up into two tasks: a master and a worker. A single copy of the master task is placed on the root processor, identical copies of the worker task are placed on all the others. The master task splits the work load into a number of smaller job units, and sends them out onto the network to be handled by the worker tasks. The load balancer controls all traffic on the network, ensuring that jobs sent out by the master task are distributed evenly throughout the network of worker tasks.

Not all programs can be implemented using this model, but it is ideal for the Root program, which is shown in Figure 4. Data is passed data back and forth between the master and worker tasks through the `PACKET` structure, using the `net_send()` and `net_receive()` functions. The master task sets the data to be processed in the `seed` field, and its square root is returned in the `root` field. As the master task cannot make any assumptions about the order in which tasks are completed, the packet number is also included within the packet, so returned data can be re-assembled into the original order. To ensure maximum throughput, the master task creates two concurrent processes, `send()` and `receive()`, to handle the creation and retrieval of these packets.

The benchmark timings in Figure 2 show that the 3L versions of Root are the fastest in the table, reflecting the low overhead imposed by the compiler and network manager. In addition, the timings remain consistent as the communications traffic increases. The benchmark programs were also run on a single transputer, and the quadputer benchmarks were consistently three times as fast as the single transputer counterparts.

Helios

Helios is a UNIX-like multi-tasking operating system, developed by Perihelion Software. Its development is closely linked to that of the Atari Transputer Workstation (ATW), née the Abaq, which was developed by its sister company, Perihelion Hardware: the ATW uses Helios as its native operating system. However, due to a number of commercial and technical problems, the ATW has made little impact on the workstation market. Helios-PC, which was used in this review, is the development system for PC hosted transputer cards.

Helios is a distributed operating system; it has no central services supporting the whole system. Its design is based on the familiar client-server model, where application tasks request services from system

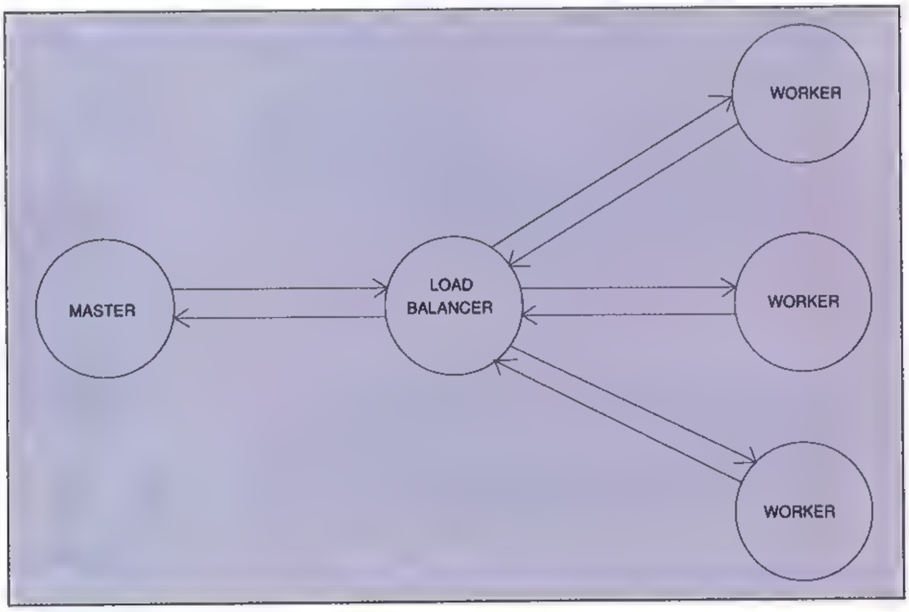


Figure 3 - The Farm model

the TRANSPUTER COMPILERS WORLDWIDE

For more information,
ask your transputer hardware supplier,
or write directly to:

3L Ltd.
Peel House
Ladywell
Livingston
EH54 6AG
Scotland

Tel 0506 415959
Fax 0506 415944

TRANSPUTER VILLAGE PC USER SHOW OLYMPIA 19-21 JUNE

IN INDUSTRY
IN EDUCATION
IN FINANCE

In thousands of sites where transputers are being applied today.

3L Parallel C, Fortran and Pascal compilers give PC-Hosted transputers transparent access to DOS files and graphics-from the DOS prompt, without special languages or operating systems. Developers benefit from the Tbug interactive debugger. And end users see transputer power in a familiar environment.

Of course, choosing the industry standard also gives you the widest range of compatible 3rd party products. Training courses, assemblers, Rowley Modula-2, libraries for graphics, Microsoft Windows, NAG functions, vector and image processing.

PARALLEL PROCESSING
MADE EASY

CIRCLE NO. 893

Another New Product from the makers of Soft-ICE & MagicCV

ANNOUNCING BOUNDS-CHECKER

Gives you the protection of a protected operating system under MS-DOS.

AUTOMATIC PROTECTION

Flush out those Nasty pointer problems and other out-of-bounds memory accesses — AUTOMATICALLY.

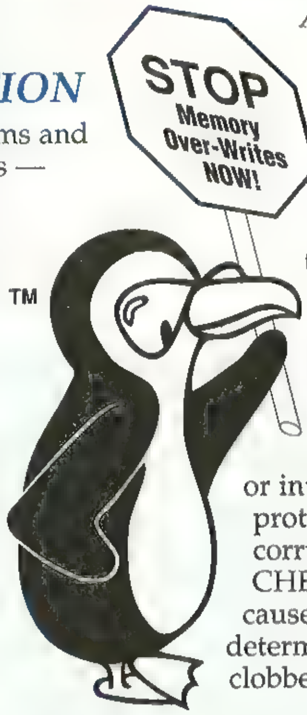
Each time you make a change to a program, run BOUNDS-CHECKER while testing the new code. If you accidentally access out-of-bounds memory, BOUNDS- CHECKER will pop up displaying the offending SOURCE LINE. And your program runs at full speed.

Increase Productivity During Development, Improve Reliability During Q/A

You can run BOUNDS-CHECKER while testing your program. There are no additional steps to your testing cycle, but you can feel secure when the program has passed through BOUNDS-CHECKER with no reported problems.

Many over-write problems and other out-of-bounds memory accesses do NOT show up during normal testing. An out-of-bounds memory location may be modified, but that particular location doesn't happen to be important at the time. Once the program is in the field and a certain network is loaded or a certain T&SR or device driver is loaded, that memory location suddenly becomes very important... AND THE SYSTEM CRASHES.

You can prevent these problems by making BOUNDS-CHECKER a standard part of your testing procedure.



Finds out-of-bounds memory accesses — AUTOMATICALLY.

HOW IT WORKS

BOUNDS-CHECKER uses the 80386 virtual machine technology to provide real-time memory protection. In addition BOUNDS-CHECKER uses the symbolic information output by your compiler to differentiate CODE and DATA. When your program is running, BOUNDS-CHECKER protects the program's CODE and all memory outside your program. When an MS-DOS system call, BIOS call or interrupt occurs, BOUNDS-CHECKER protects the system software from corrupting your code. So, BOUNDS-CHECKER will not only detect problems caused by your program, it will also determine if a T&SR or other program is clobbering you.

Enormous Productivity Gain

Don't spend days or even weeks chasing down one of those subtle memory over-write problems. Use BOUNDS-CHECKER to pinpoint the problem in *seconds*.

Don't RISK a Recall

Don't take the unnecessary RISK that your program is unknowingly clobbering an out-of-bounds memory location. BOUNDS-CHECKER was developed to find these potential time bombs quickly, painlessly and AUTOMATICALLY!

$249.00

Requires 80386 PC.
MS-DOS is a trademark of
Microsoft Corporation.

Nu-Mega
TECHNOLOGIES

CALL TODAY (603) 888-2386 or FAX (603) 888-2465
P.O. BOX 7607 ■ NASHUA, NH ■ 03060-7607 ■ U.S.A.

CIRCLE NO. 894

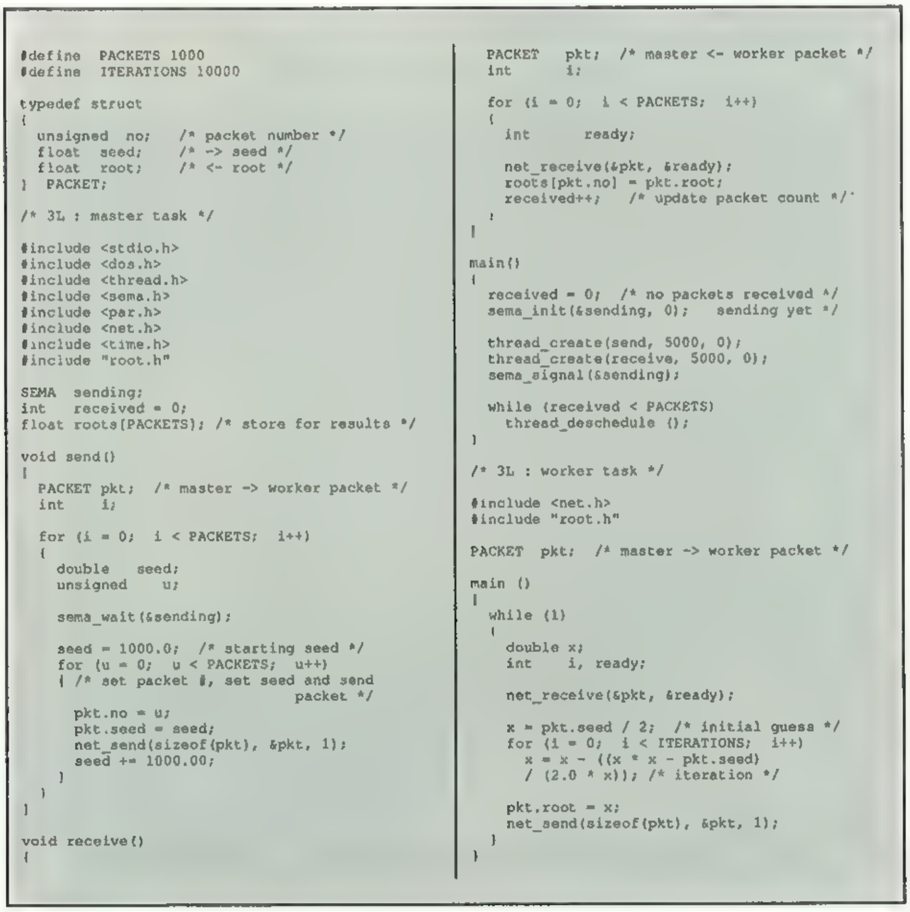
provided server tasks. The Helios nucleus, along with any servers, such as file handlers, window managers, etc, must be present on each processor in the network. A comprehensive description of Helios appeared in .EXE Magazine, January 1988.

Installing Helios-PC is a fairly straightforward task, though configuring it for the Quadputer transputer board, which was used to perform all the tests described in this article, proved slightly more difficult, as no suitable driver was originally included in the package. Once the correct driver, plus an upgrade to V1.1a, had been obtained, no further problems were encountered. This upgrade highlights one of the main problems with Helios, namely it is a product that is still developing. Earlier versions of Helios had a reputation for being unstable, though many of these problems seemed to have been cured.

The Helios environment includes a shell very similar to the standard UNIX C shell. There is also a copy of the EMACS text editor, and a spartan selection of UNIX-like utilities. The Helios development system comes with a C compiler and assembler. The compiler supports many ANSI features, and the libraries are UNIX compatible, and designed to comply to the proposed POSIX standard. A C source level debugger is also available.

There are three approaches to developing programs under Helios. The first is the traditional sequential model, in which a program runs as a single task under Helios. Thus, programs ported from environments such as UNIX or MS-DOS can be run under Helios with few changes. The second approach is to a run an application program as a set of sequential tasks communicating through pipes. An example of this might be a compiler that can run as three tasks; preprocessor, code generator and linker, each task passing its output onto the next task. A single processor operating system would timeslice between these tasks, but Helios can allocate each one to a separate processor, so that they run in parallel. The final approach is to develop the program using parallel algorithms, explicitly dividing the program into a number of discrete tasks which distribute the program's workload.

Helios offers fairly course-grained parallelism: the smallest unit of execution that it deals with is a task. This technique can offer significant speed improvements for a small amount of programming effort. However, programs can still be developed using finer-grained parallelism, if required.



```
#define  PACKETS 1000
#define  ITERATIONS 10000

typedef struct
{
  unsigned  no;    /* packet number */
  float  seed;     /* -> seed */
  float  root;     /* <- root */
}  PACKET;

/* 3L : master task */

#include <stdio.h>
#include <dos.h>
#include <thread.h>
#include <sema.h>
#include <par.h>
#include <net.h>
#include <time.h>
#include "root.h"

SEMA  sending;
int   received = 0;
float roots[PACKETS]; /* store for results */

void send()
{
  PACKET pkt;  /* master -> worker packet */
  int    i;

  for (i = 0;  i < PACKETS;  i++)
  {
    double   seed;
    unsigned    u;

    sema_wait(&sending);

    seed = 1000.0;  /* starting seed */
    for (u = 0;  u < PACKETS;  u++)
    { /* set packet #, set seed and send
                                packet */
      pkt.no = u;
      pkt.seed = seed;
      net_send(sizeof(pkt), &pkt, 1);
      seed += 1000.00;
    }
  }
}

void receive()
{
  PACKET   pkt;  /* master <- worker packet */
  int      i;

  for (i = 0;  i < PACKETS;  i++)
  {
    int      ready;

    net_receive(&pkt, &ready);
    roots[pkt.no] = pkt.root;
    received++;   /* update packet count */
  }
}

main()
{
  received = 0;  /* no packets received */
  sema_init(&sending, 0);   sending yet */

  thread_create(send, 5000, 0);
  thread_create(receive, 5000, 0);
  sema_signal(&sending);

  while (received < PACKETS)
    thread_deschedule ();
}

/* 3L : worker task */

#include <net.h>
#include "root.h"

PACKET  pkt;  /* master -> worker packet */

main ()
{
  while (1)
  {
    double x;
    int    i, ready;

    net_receive(&pkt, &ready);

    x = pkt.seed / 2;  /* initial guess */
    for (i = 0;  i < ITERATIONS;  i++)
      x = x - ((x * x - pkt.seed)
      / (2.0 * x)); /* iteration */

    pkt.root = x;
    net_send(sizeof(pkt), &pkt, 1);
  }
}
```

Figure 4 - 3L version of Root

Helios uses its own language, Component Distribution Language (CDL) to define the tasks that make up an application program. A CDL definition consists of two parts; a description of how the tasks communicate with each other, and, optionally, details of the resource requirements for each task. The task force manager uses the CDL definition to map the tasks onto the network of processors. The advantage of CDL is that the application program can be defined in a way which is independent of the actual network. It is the task force manager which decides where to place the tasks, ensuring that the application program will work on a single processor or a network of processors.

CDL is designed to appear as an extension to the UNIX shell. For example:

```
ls | more
```

is actually a CDL script, which specifies that the programs `ls` and `more` are to run as two separate tasks, with the output of `ls` providing the input for `more`, as shown in Figure 5. Bidirectional pipes are also possible:

```
scroff <> filter
```

specifies two concurrent tasks, `scroff` and `filter`. `scroff` passes data onto `filter` for processing, and the output from `filter` is directed back to `scroff`, as illustrated in Figure 5. Several other similar constructs are also supported. Using these, it is possible to build quite sophisticated application programs. The optional description of resource requirements, mentioned above, is designed for tasks with specific requirements, such as a minimum amount of memory or access to a particular resource.

Rather than utilising special library calls, as with 3L's Parallel C, tasks under Helios communicate via the standard I/O library, using the `read()` and `write()` library functions. In the above CDL example, `filter` reads from standard input, and writes to standard output, and is thus unaware that it is running as a subordinate task for `scroff`. However, as `scroff` has to communicate with external tasks using its own standard input and output channels, it needs two additional channels for communications with `filter`; these correspond to the POSIX file descriptors 4 and 5.

Root Under Helios

One of the most powerful CDL constructs is the farm construct, which corresponds very closely with 3L's farm model, shown in Figure 3. A master task communicates with a number of identical worker tasks via a load balancer. The CDL script for the farm construct is

```
master [3] ||| worker
```

The '`[3]`' is a replicator, which specifies that three worker tasks are to be created. However, unlike the 3L farm model, any number of workers can be specified. As the load balancer is just another program task, the source code is provided, enabling more sophisticated and application-specific versions to be developed.

The Helios version of the Root program is shown in Figure 6. It is very similar to the Parallel C version. The `LB_HEADER` construct must be part of the packet header so that packets can be handled.correctly by the load balancer. The `size` field specifies the size of the packet, and the `control` field is used to broadcast to all the workers, in this case to issue the terminate command. Communication between tasks is through the standard `read()` and `write()` functions.

The benchmark timings for the Helios Root program run at about one tenth of the speed of the 3L versions (Figure 2), indicating the overheads imposed by the task force manager. To ensure comparability, the benchmarks were timed using three worker tasks.

Conclusion

The 3L and Helios systems offer somewhat different approaches to program development under C. The 3L Parallel C compiler is designed to integrate into the normal MS-DOS environment, whereas Helios creates an entirely new UNIX-like development environment. They each have their own strengths and weakness - to some extent it's a trade-off between the amount of programming effort veusus performance gain - but both let programmers get at the power of the transputer. EXE

David Gristwood is in charge of new product development at Byline Software, which specialises in micro and mini computer software development. He can be contacted on 091-386 0286, or on CIX as 'scroff'.

Product details:
The Parallel C and Tbug software packages cost £750 and £220 respectively, from 3L Limited (0506 415 959).
The Helios operating system (PC version) with C compiler costs £750, the Helios Source Debugger costs £485. Helios can be obtained from Distributed Software Ltd (0454 612 777).

Figure 5 - Communication of CDL tasks

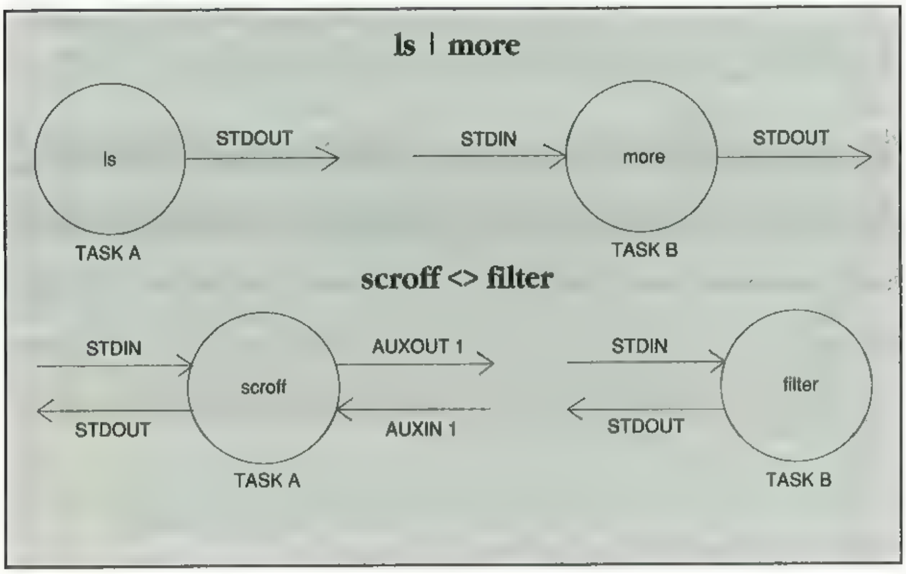


Figure 6 - Helios version of Root

```
#define  PACKETS 1000
#define  ITERATIONS 10000


typedef struct
{
  LB_HEADER header; /* load balancer header */
  unsigned  no;     /* packet number */
  float  seed;      /* -> seed */
  float  root;      /* <- root */
}  PACKET;



/* HELIOS : master task */


#include    <helios.h>
#include    <stdio.h>
#include    <stdlib.h>
#include    <posix.h>
#include    <sem.h>
#include    <nonansi.h>
#include    "newlb.h"
#include    "root.h"


float    roots[PACKETS];
Semaphore   finished;  /* semaphore:finished */


void     send()
{
  PACKET   pkt;  /* master -> worker packet */
  double   seed;
  int      i;


  pkt.header.control = 0;  /* normal packet */
  pkt.header.size = sizeof(pkt)
- sizeof(LB_HEADER);


  /* send each packet */


  seed = 1000.0;  /* initial seed */
  for (i = 0;  i < PACKETS;  i++)
  {
    pkt.no = i;
    pkt.seed = seed;
    write(5, (byte *) &pkt, sizeof(pkt));
    seed += 1000.00;
  }
}


void     receive()
{
  PACKET   pkt;  /* master <- worker packet */
  int      i;


  /* wait for each packet */


  for (i = 0;  i < PACKETS;  i++)
  {
    read(4, (byte *) &pkt, sizeof(pkt));
    roots[pkt.no] = pkt.root;
  }

  Signal(&finished);  /* signal finished */
}


int      main()
{
  LB_HEADER   terminate;
  time_t   t;
  unsigned u;


  InitSemaphore(&finished, 0);semaphore */
  Fork(5000, &send, 0);
  Fork(5000, &receive, 0);    ceive() thread */
  Wait(&finished);


  terminate.control = LB_MASTER
          + Fn_Terminate;  /* terminate flag */
  terminate.size = 0;      /* no data portion */
  write(5, (byte *) &terminate,
  sizeof(terminate));  /* terminate */
}



/* HELIOS : worker task */


#include    <helios.h>
#include    <stdio.h>
#include    <stdlib.h>
#include    <posix.h>
#include    <sem.h>
#include    <nonansi.h>
#include    "newlb.h"
#include    "root.h"


int      main ()
{
  PACKET   pkt1, pkt2;
  double   x;
  unsigned    u;


  while (TRUE)
  {
     read(0, (byte *) &pkt1, sizeof(pkt1));
     if ((pkt1.header.control & LB_FN)
     == Fn_Terminate)
        exit(0);  /* terminate */


     x = pkt1.seed / 2;  /* initial guess */
     for (u = 0;  u < ITERATIONS;  u++)
        x = x - ((x * x - pkt1.seed)
                     / (2.0 * x));


     pkt2.root = x;  /* store result */
     pkt2.no = pkt1.no;
     pkt2.header.control = 0;
     pkt2.header.size = sizeof(pkt2)
     - sizeof(LB_HEADER); /* size */
     write(1, (byte *) &pkt2, sizeof(pkt2));
  }
}
```

POWER AND FLEXIBILITY

BRIEF 3.0 increases your productivity for just £159.00*

*VAT extra, FREE delivery

BRIEF has established itself as the leading programme editor on the market today. Now, with the launch of the latest enhanced version – BRIEF 3.0 – it offers even better productivity.

ENORMOUS UNDO POWER

BRIEF 3.0's enhanced UNDO capability takes the fear out of programming. BRIEF 3.0 allows you to undo any command that affects the cursor or the text (including global changes) up to 300 times, even back to the beginning of your session. It's the safety net you need to program with confidence.

REAL FLEXIBILITY

There's built-in support for 36 languages including Assembler, BASIC, C, COBOL, dBase, FORTRAN and Pascal. And thanks to its complete macro language you can make BRIEF 3.0 fit your own programming style with ease: change key assignment, commands, prompts, colours – even the indenting styles.

BRIEF 3.0 – new benefits

- With our new C-like macro language, CBRIEF, C programmers can write macros on the fly and not have to switch between C and a different macro language
- The new source level debugger works with both the original BRIEF syntax and the new CBRIEF macro language
- Maximum lines per file has been increased to 4 billion
- We've added to and improved template editing and smart indenting for ADA, Cobol, Basic, Fortran, Modula-2, Pascal and C
- We've added editable, multiple keystroke macros that can be saved and restored
- You can even save and restore your window set-ups

Plus the features that made BRIEF famous

- Flexible windowing
- Unlimited number of files
- File size limited only by desk space
- Regular expression search and replace

No other editor puts this much power and flexibility in your hands. So if you're looking for extra productivity purchase your copy of BRIEF 3.0 today.

Solution Systems

SOLUTION SYSTEMS are the UK distributors for BRIEF 3.0. When you buy from us you get the authorised UK version of BRIEF 3.0 dual media (3½" and 5¼") plus bonus macro diskette.

Special phone line for Access/Visa card holders. Save time by phoning **0763 73457** between 9.00 and 6.00 Monday to Friday to place your order.

N.B. Official purchase orders
from local authorities,
PLCs and educational establishments
are accepted by post or by Fax.
Fax no: 0763 73460

BRIEF 3.0 The Programmer's Editor

Send to: Solution Systems UK, The Red House, 84 High Street, Buntingford, Hertfordshire SG9 9AJ

Please rush me my copy of BRIEF 3.0

□ I enclose a cheque to the value of £182.85 (£159.00 + V.A.T.) crossed and made payable to Solution Systems UK. **Orders despatched same day.**

NAME ADDRESS

POSTCODE

□ Please charge my Access/Visa* account the sum of£182.85 (£159.00 + V.A.T.)

Card No ______ **Expiry Date** ______ **Signature** ______

EXE 5.90

*Delete as appropriate

CIRCLE NO. 895

Razor sharp

Occam, the language that only gets a capital letter when it's at the beginning of a sentence, has an exotic reputation. Unfair! cries John Wexler.

Let's plunge in at the deep end. Take a look at my solution to the Triangle Problem, shown in Figure 1. There you are: a fragment of an occam program. There are plenty of recognisable features - the words `IF`, `AND` and `FOR`; strings enclosed in double quotes; procedure calls, with parameters enclosed in parentheses; arrays subscripted using `[` and `]`; and declarations, each followed by a colon `:`, prefixed to the body of the text. In fact, it probably does not look as strange as you expected of 'that odd language that they use with transputers'. Clearly it is a tidily laid out program, in a familiar imperative language, and not at all weird like Prolog (declarative logic) or Eiffel (object-oriented) or SML (functional) or LISP (list processing). 'No surprises, no complications' are the guiding principles of occam - and this is how it came by its name. The fourteenth-century English philosopher, William of Occam, coined the motto which is known as Occam's Razor: *non sunt multiplicanda entia preater necessitatem* ('never multiply entities unnecessarily' ie 'keep it simple').

Origins

These principles were not selected for fun, academic interest, or even love of Latin. They were essential for the commercial success of a very major project: the development and marketing of the transputer. To attract attention, the product had to be revolutionary, it had to be targeted at several different markets, and it had to work better than its competitors. The 'revolutionary' aspect would lie in its unprecedented support for concurrency and multiprocessor working. Diversity of marketing sectors meant that it had to suit applications from single processor embedded systems to 1000 processor supercomputers. As for quality, it had to be very good indeed, and it had to be right first time. All of that meant that it could not be left solely in the hands of hardware engineers. Alongside the inputs

LYNNE CHAPMAN

from manufacturing technology, market analysis, logic design and so on, the design process had to accept a major contribution from the software side: system designers, compiler writers and, above all, theoreticians.

The chosen approach was to decide first which functions were required, and then to devise a machine-independent language which could express those very precisely. Only after that would the processor be designed, to implement that language as efficiently as possible. In this way, it was hoped to embody 'concept' in silicon with guaranteed correctness. The language, of course, became occam.

Now, up to this point, the idea of end users actually writing real applications in occam has not been mentioned; but a language which is simple, precise and susceptible to analysis is rather attractive for serious programming, especially if the user is also having to come to grips with the novel and tricky ideas of concurrency.

Inmos marketed the transputer and occam as a package. With hindsight, it is easy to see that this put some people off transputers, and made it easy for others to ignore occam. Fortunately, however, both have survived, and things look different now. Ironically, the turn-around has been much helped by the creation of transputer compilers for other languages, such as C, FORTRAN and Pascal. Occam depends on the transputer; the transputer needs to support conventional languages to succeed; hence occam's survival is partially attributable to the likes of C.

The language

Occam is procedural imperative language with variables to which you can assign values (using the := symbol), sequences of instructions to be obeyed one after another, procedures and functions, and so on. It handles data of integer, Boolean, byte (character) and real types. You can have literals, named constants, variables and arrays (a string is handled as an array of bytes). It provides extra features for concurrent and real-time programming: primarily for specifying that several processes can be performed concurrently rather than in sequence, and for passing messages between them. There are timers for controlling delays. All in all, there are very few basic ideas, and they are all very simple. Thus the official definition of occam is exceptionally compact, and even lucid. However, the ideas can all be combined, limited only by the programmer's ingenuity, to make up a comprehensive set of facilities for general as well as concurrent programming.

To many programmers in C and Pascal, the oddest thing about occam is that you cannot write it in free format. The indentation of each line of text is very important, and affects the meaning of the program. These indentation rules actually ensure that a program's layout is significant to the human reader as well as to a compiler. In effect, the programmer is forced to follow what, in Pascal, would be 'good practice'. Furthermore, there is a 'folding editor' for occam, which understands and supports these 'indentation rules', so it is not difficult to follow them. Incidentally, this is one of many cases where what is 'good practice' in other languages is made obligatory in occam.

At the heart of the occam is the concept of a 'process'. An example of a very simple process is the assignment of a new value to a variable is. Just as other languages have simple 'statements' which can be grouped

Figure 1 - The Triangle Problem

```
[3]INT side:
a IS side[0]:
b IS side[1]:
c IS side[2]:
VAL OTHERWISE IS TRUE:
SEQ
  Readn (keyboard, a)
  Readn (keyboard, b)
  Readn (keyboard, c)
  SEQ i=0 FOR 3
    Writef2 (screen, "%i5*n", [side[i]])
  IF
    IF i=0 FOR 3
      side[i] > (side[(i+1) REM 3]+side[(i+2) REM 3])
        Writes (screen, "This is not a triangle.*n")
    (b=c) AND (c=a)
      Writes (screen,
            "This is an equilateral triangle.*n")
    IF i=0 FOR 3
      side[i]=side[(i+1) REM 3]
        Writes (screen,
              "This is an isosceles triangle.*n")
    OTHERWISE
      Writes (screen, "This is a scalene triangle.*n")
```

together to make 'compound statements' - perhaps between BEGIN and END - so occam can group several processes into one larger process (and so on, to any desired depth). As in Pascal, you can ask for the processes to be performed one after another in a particular order. You write the word SEQ (for 'sequence') instead of BEGIN, followed by the processes in order. C programmers, who snear at Pascal for its surfeit of BEGINs and ENDs, will be impressed by the fact that the indentation of source text shows which processes are controlled by the SEQ, so there is no need for an END.

Concurrency

There are, however, other ways to group processes. You can ask for several of them to be performed in parallel, simply by writing PAR instead of SEQ. That means that they can be done in any order, or perhaps by doing a bit of one and a bit of another until they have all been completed. If you have multiple processors, of course, they can be genuinely concurrent. The question then arises of how information can be shared between concurrent processes. In occam, they cannot share variables, but they can pass data from one to another through 'channels'. In fact the only primitive processes, apart from assignment, are the message-passing 'input' and 'output'.

```
CHAN OF INT forward:
PAR
  INT x:
  SEQ
    x := 12
    forward ! x
  INT y, z:
  SEQ
    forward ? z
    y := z + 1
```

Here we have two processes, to be performed concurrently (they are written after the PAR). Each of the two consists of a SEQuence of smaller processes. The first contains an assignment, followed by an output, denoted by the ! symbol. The other consists of an input (marked by ?) followed by an assignment. On the first line, `forward` is declared as a channel which can carry messages consisting of single integers. `forward` is in scope through the whole of the 'parallel' construct. Declarations are prefixed to a process, and apply to the whole of that process, but no further. Similarly, the single integer variable x is in scope for the three lines beneath its declaration.

What will happen when we run this? The interesting part is the input/output pair. When one process does `forward ! x` and the other does `forward ? z`, the value of x is transmitted from one process to the other and stored in z. The effect is like an assignment `z := x`. However, we could not write z := x, because x belongs to one process and z to the other, and processes may not share variables. We could, perhaps, describe the effect of the input and the output as a 'distributed assignment'.

The first SEQ forces `x := 12` to take place before the output `forward ! x`. Similarly, `forward ? z` will happen before `y := z + 1`. But will `x := 12` start before or after `forward ? z`? It's impossible to say! This is one of the difficulties with concurrency - you never know exactly in what order things will happen. Then how can I claim that occam is precise and unambiguous? It turns out that it simply does not matter which one starts first. Regardless of when `forward ? z` starts, it cannot finish until the other process has done `forward ! x`.

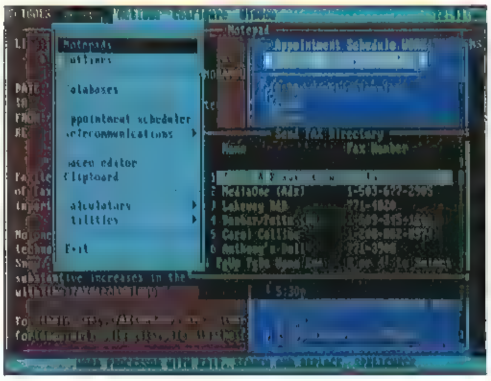
There is a momentary synchronisation between the two processes. Whichever process comes first to attempt an input or an output, that process will be suspended until the other participant performs the complementary input or output. (Putting it another way, the channel does not buffer messages.)

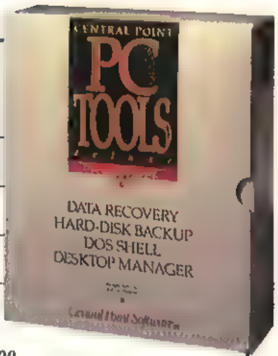
Multi-channel

In systems of many concurrent processes, it often happens that one process has to communicate with a number of others. This is easily done in occam, by using a different channel for each line of communication. In real systems, however, there is a complication: a process which has to be able to handle input from several others may not 'know' which one is going to offer input next. It must, therefore, attempt an input for more than one channel. It cannot use the ? operator in a SEQ compound process, so occam provides for this with a construction marked by the word ALT (for 'alternative').

```
ALT
  fromTom ? a
    x := a + 1
  fromDick ? a
    x := a - 1
  fromHarry ? b
    SEQ
      x := x + b
      a := 1
```

After ALT, you list all of the possible inputs, following each one with the process that is to be performed if that input is received. The whole ALT process will accept just one of the inputs, and perform the one corresponding process. Which of the inputs will be chosen? Whichever one becomes available first. If more than one is ready when the ALT is started, then any of them will be chosen; the others will remain available, and can be accepted some time in the future.





The strange thing about ALT is that it introduces a non-determinacy into the language. If a program contains even one ALT, it is impossible to predict how it will behave! And do I still claim that occam is precise and unambiguous? I do - but not, perhaps, in the sense which sequential programmers would like. A degree of non-determinacy is unavoidable in concurrent systems. Occam confines it to one very clear and simple part of the language, and gives you the best chance of understanding it and coping with it - it is as good as you can get in a difficult situation.

Besides SEQ and PAR and ALT, occam offers two more groupings of processes - WHILE and IF. On the whole, these behave pretty much what you would expect, although IF does differ from the norm. In most languages, the basic IF construction says 'IF condition THEN do this ELSE do that'. You build up more elaborate control structures by compounding 'IFs within IFs'. In occam, IF can be followed by a list of as many conditions as you please, with a process corresponding to each condition. The list of conditions is scanned, until one is found which is TRUE; the process which corresponds to that condition is performed. This means that there is less need for nested IFs. However, you do have to write 'IF condition THEN do this ELSE do nothing' when you just mean 'IF condition THEN do this'. It looks like this:

```
IF
  a=0
    x :- x + 1
  TRUE
    SKIP
```

SEQ, PAR, ALT, and IF can all be followed by any number of processes. This leads to an interesting generalisation of the loop, which is called a 'replication' in occam:

```
SEQ i=0 FOR 3
  x := x + p[i]
```

which is equivalent to

```
SEQ
  x := x + p[0]
  x := x + p[1]
  x := x + p[2]
```

which is just like a conventional loop. You can also write

```
PAR i=0 FOR 3
  out[i] ! id
```

to do three processes in parallel - in this case, sending the same data out through three different channels (out is an array of channels). Naturally, you can apply replication to IF in exactly the same way, like this:

```
IF i=0 FOR 3
  name[i]<>pattern[i]
    mismatch := i
```

means

```
IF
  name[0]<>pattern[0]
    mismatch := 0
  name[1]<>pattern[1]
    mismatch := 1
  name[2]<>pattern[2]
    mismatch := 2
```

which scans two arrays, and assigns the index of the first non-matching elements to mismatch. In other languages, this would be a construction where you would be tempted to use GOTO to get out of the loop when the first mismatch had been found; or you would find some other more of less subtle mechanism to express what is really quite a simple requirement. None of that is necessary in occam; in fact, given that there is also WHILE construction (and a CASE), occam needs no GOTO at all. It simply is not in the language!

The official definition of occam is exceptionally compact, and even lucid

The absence of GOTO is one feature which makes it possible to analyse occam programs with logic. Another is a total ban on side-effects of all kinds. All storage allocation is on the stack; there is no heap, and no equivalent of C's malloc(). Pointer variables are not used, but there is something called an 'abbreviation', which serves some of the same purposes without any risk of such notorious pointer-related bugs as 'dangling references' and 'conflicting usage'. Occam is extremely rigorous about data types, and will never do a type conversion unless the user explicitly asks for it. There are comprehensive rules forbidding potentially risky sharing of data channels between concurrent processes. All of this means the occam programs can be extensively checked during compilation, saving a great deal of debugging later.

Restrictions

Occam 2 has two principle restrictions, by comparison with (say) Pascal: it has no recursion, and all storage allocation is static (that is, all addresses can be calculated at compile time, or at least before the program is loaded). There are two reasons for this. First, it is very hard to do recursion and dynamic storage allocation on behalf of a number of processes which are running concurrently and asynchronously, and current transputers give no hardware support to make that any easier. Second, if one allowed them, 'running out of store' could happen at any time, unpredictably, and that would make it impossible to guarantee the behaviour of real-time safety-critical systems.

Occam also lacks 'unions', 'structures', 'records' or the like and in fact it has no way for the programmer to define new 'types' of data.

Now we can look back at the 'triangle program'. The main program is controlled by SEQ. Within that, there are three procedure calls on Readn to accept the input, followed by a replicated SEQ - a loop - for printing out the three given numbers, and finally an IF to apply the tests. This IF has four separate tests for the four conditions: 'not a triangle', 'equilateral', 'isosceles' and 'scalene'. Two of those conditions - 'not a triangle' and 'isosceles' - are themselves implemented by replicated IFs . There are abbreviations a, b and c for the elements of the side array which illustrate the use of this technique. I also use the abbreviation (sic) OTHERWISE for TRUE to clarify the final condition in an IF process. EXE

John Wexler works at the Edinburgh University Computing Service, where he is connected with the Edinburgh Concurrent Supercomputing Project, which has a Meiko Computing Surface comprising of around 500 transputers.

If you want to find out more about occam, he recommends his own book Concurrent Programming in occam 2 (pub Ellis Horwood, ISBN 0-7458-0394-6) and also the occam 2 Reference Manual (pub Prentice Hall, ISBN 0-13-629312-3). If you prefer a really heavy read, he suggests getting a book on Ada instead.

If you have a favourite, non-mainstream language, and you would like to write about it for 'The Third Side' series, please write to The Editor, at the address given on page 2, for a copy of the guidelines.

CALL FOR FREE 80 PAGE CATALOGUE

ASSEMBLERS

386 ASM Link & Locate	£435.00
MASM v.5.1	£89.00
OPTASM	£79.00
Turbo Assembler/Debugger	£89.00

BASIC

Microsoft Professional BASIC	£320.00
BASIC Interpreter	£215.00
Quick Basic *U.K*	£60.00
True Basic	£65.00
Turbo Basic	£65.00

C

Lattice C Compiler v.6.0	£160.00
Microsoft C v.5.1	£250.00
Watcom C Optimizing	£245.00
C-terp	£189.00
C86 Plus	£310.00
Eco C88 with dev. library	£55.00
Express C	£60.00
High C-286	£375.00
High C-386	£560.00
Instant C	£310.00
Instant C/16M	£495.00
NDP C	£375.00
Quick C *U.K*	£65.00
Turbo C v.2.0	£89.00
Turbo C Professional	£159.00
UserSoft/C	£150.00
Zortech C Compiler	£45.00

C++

Advantage C++	£310.00
Guidelines C++ with book	£185.00
Intek C++ 80386	£310.00
Zortech C++ Compiler v.2.0	£120.00
Zortech C++ Dev. Edition	£270.00

COBOL

acucobol MS-DOS	£900.00
acucobol Net-Bios	£1250.00
acucobol XENIX-286	£1500.00
acucobol XENIX-386	£2000.00
acucobol OTHERS	CALL
Microsoft COBOL	£565.00
Real Menu	£210.00
RealCICS	£750.00
Realia COBOL	£750.00
RM/Cobol	£575.00
RM/Cobol-85	£750.00

dBASE

Clipper Version 5.0 U.K. *New*	£325.00
DB Fast/PLUS	£155.00
DB Fast/WINDOWS	£155.00
dBASE III Plus	£395.00
dBASE IV	£395.00
dBMAN V	£120.00
dBXL Diamond	£155.00
dBXL/Lan	£375.00
Foxbase+ v.2.10	£245.00
Foxbase+/386	£375.00
Foxbase+/LAN	£375.00
Foxbase+ Runtime	£435.00
Foxbase+ SCO XENIX	£495.00
Quicksilver Diamond	£375.00

DATABASE MANAGEMENT

Advanced Revelation	£595.00
AskSam	£185.00
Clarion Personal Developer	£99.00
Clarion Professional	£436.00
Omnis Quartz	£480.00
Opus One	£240.00
Oracle for 1-2-3	£120.00
Paradox 2.0	£439.00
PFS Professional File	£149.00
R:BASE Ext. Report Writer	£179.00
R:BASE for OS/2	£545.00
R:BASE for DOS	£439.00
Reflex	£89.00

FORTRAN

Microsoft Fortran v.5.0 *U.K*	£285.00
Fortran-F77L	£299.00
NDP FORTRAN-386	£360.00
ProFortran	£235.00
RM/Fortran	£375.00
SVS Fortran	£435.00

OTHER LANGUAGES

ACTOR	£310.00
ACTOR Language Extension	£60.00
CCSM Mumps Single User	£65.00
PC Logo	£90.00
RPG II Complete	£995.00
Smalltalk/V	£65.00
Smalltalk/V 286	£129.00
Snobol 4+	£60.00
Trilogy	£60.00

PASCAL

Microsoft Pascal	£189.00
Oregon Pascal-2	£149.00
ProPascal	£235.00
Metaware Professional Pascal	£560.00
Metaware Prof. Pascal 286	£375.00
Turbo Pascal v.5.5 with objects	£89.00
Turbo Pascal Professional	£159.00

PROLOG

Arity Combination Pack	£685.00
Expert System Dev. pack	£185.00
Intelligence Compiler	£299.00
MicroPROLOG Professional	£425.00
MProlog	£299.00
Professional Prolog II Plus	£899.00
Professional Prolog	£545.00
Prolog Advanced Toolkit	£90.00
Prolog Compiler	£120.00
Prolog Compiler/Interpreter	£399.00
Prolog-86 Plus	£150.00
SQL Development Pack	£185.00
Turbo Prolog	£75.00

AI AND OBJECT ORIENTED

C-Talk	£95.00
C+ Objects Vol 1 with Source	£275.00
CxPERT	£245.00

COMMUNICATIONS

AdComm	£185.00
C ASYNCH MANAGER	£110.00
Communications Library	£145.00
Comm Pak with Breakout	£195.00
Multi Comm	£89.00
Net Lib	£125.00
NET-TOOLS	£95.00
SilverComm	£120.00
Comm Plus	£60.00
Turbo ASYNCH PLUS	£80.00

FILE MANAGEMENT

asmTREE	£245.00
B-Tree with Source	£110.00
B+Tree for C	£49.00
AXOS	£245.00
Btrieve v.5.01 Novell	£155.00
Btrieve OS/2	£370.00
Btrieve Multitasking	£370.00
Btrieve/N v.5.01	£370.00
Btrieve XENIX	£370.00
C Index+	£245.00
c-tree	£245.00
Code Base 4	£185.00
db_FILE/RETRIEVE	£375.00
d-tree	£310.00
dBC III	£160.00
dBC III Plus	£320.00
RealDL/I	£599.00
WKS Library	£125.00

GENERAL

ASMLIB	£89.00
C Tools Plus	£79.00
C Utility	£120.00
C*Lib C Function Library	£120.00
Clipper toolbox	£239.00
DB/RAS	£125.00
dBase Program Utility	£55.00
dBase Program Utility Vol 2	£55.00
dBase Tools For C	£55.00
Finally	£60.00
Functions with source	£125.00
Integrated Dev. Libraries	£90.00
Pascal Toolkit	£90.00
Pascal Tools 1	£75.00
QuickPak Professional	£90.00
Quick Tools Library	£79.00
Spindrift Library	£90.00
Super Functions	£160.00
Tom Rettig Help Combo	£75.00
Tom Rettig Library	£60.00
Turbo C TOOLS	£79.00
Turbo Plus	£60.00
Turbo PowerTools Plus	£79.00
Turbo Functions	£69.00

GRAPHICS

dBase Tools for C Graphic	£55.00
dGE v.3.0 *NEW*	£125.00
EGA Toolkit	£149.00
Essential Graphics	£180.00
Flipper	£120.00
FontWINDOW/Plus	£170.00
GEOGRAPH Professional	£239.00
Grafmatic CGA & EGA	£135.00
GraphiC	£245.00
GSS Graphics Dev. Toolkit	£389.00
GSS Kernel	£299.00
GSS Metafile Interpreter	£179.00
HALO-88	£245.00
HOOPS	£360.00
HOOPS 32	£349.00
MetaWINDOW/Plus	£169.00
scLASERplus	£299.00
SilverPaint	£60.00
SilverPak	£179.00
Slate	£215.00
Universal Graphics Library	£130.00

OP. SYSTEM ENHANCEMENTS

C Sharp	£300.00
DESQview API C Lib/Ref	£125.00
Multi C	£89.00
Multi Dos Plus	£60.00
OS/2 Programmer's Toolkit	£215.00
OS/286	£299.00
OS/386	£299.00
Resident C	£120.00
Sapiens V8	£180.00
Synergy Toolkit	£239.00
Time Slicer	£179.00
VMFM	£115.00
Wendin PC UNIX	£60.00
Wendin PC VMS	£60.00
Windows Development Kit	£295.00

QUERY

CBTREE/SQL	£239.00
CQL Query System	£239.00
dQUERY/Lib	£179.00
db_RETRIEVE	£239.00
XQL	£480.00
Xtrieve PLUS	£360.00
Netware SQL	£385.00

REPORT

CB/IQ Report	£139.00
Coreports	£179.00
Focus Report Writer	£210.00
r-tree	£179.00
R&R Relational Reportwriter	£95.00
R&R Clipper Module	£49.00

TEXT SCREENS

Animator COBOL	£120.00
Application Manager	£450.00
C-Worthy v.1.1	£160.00
C-Worthy + Forms	£240.00
C-Worthy + Forms + Source	£395.00
C-Worthy All C compilers	£480.00
C-Worthy Forms only	£80.00
C-Worthy Source code only	£155.00
C-Worthy OS/2	£240.00
C-Worthy + Forms for OS/2	£320.00
C-Worthy + Forms + Source	£480.00
C-Worthy OS/2 & MS-DOS	£315.00
C-Worthy + source	£795.00
Curses	£150.00
Curses Window Dev. Kit	£75.00
Data & Windows	£149.00
DataWindow	£179.00
dWindows for db3	£60.00
Face-it	£60.00
Flash-up w/Toolbox	£85.00
Forms II	£179.00
HI-SCREEN XL	£89.00
Multi Windows	£179.00
On-Line Help	£90.00
PANEL PLUS	£299.00
PC Forms	£90.00
Pop Screen	£55.00
POWER SCREEN	£79.00
Pro Screen	£60.00
SCREENIO	£245.00
Screen Sculptor	£75.00
ScreenStar	£120.00
Screenplay	£110.00
SoftCode	£120.00
SoftCode Template	£30.00
Softscreen/Help Source	£150.00
Sourcewriter	£605.00
UI Programmer Dev. Edition	£345.00
V Screen	£39.00
VCScreen	£90.00
Vitamin C	£135.00
VLib with Source	£90.00
Vermont Views	£245.00
Z View	£149.00

DEBUGGERS & DISASSEMBLERS

386 Debug	£125.00
Breakout-II	£79.00
D'Scope 2	£159.00
dBug for dBase	£125.00
DIS*DOC	£79.00
Periscope II	£110.00
Periscope II-X	£90.00
Periscope III	£875.00
Periscope I Rev 3	£435.00
Periscope IV (16 MHz)	£1245.00
Periscope IV (20 MHz)	£1435.00
Periscope IV (25 MHz)	£1625.00
Plus Board with 512K	£310.00
Plus Board with 0K	£220.00
Serial Test	£310.00
Soft Probe II/TX	£245.00
Soft-X-plore	£79.00
Sourcer + BIOS Pre-Processor	£89.00

DOCUMENTORS

ASM Flow	£60.00
Clear+ for dBASE	£125.00
Clear+ for C	£125.00
dAction	£55.00
dAnalyst	£139.00
Design/2.0	£215.00
dFLOW	£89.00
Documenter	£179.00
Flow Charting II Plus	£139.00
FoxToolBox	£179.00
PolyDoc	£169.00

EDITORS

BRIEF v.3.0 *NEW*	£149.00
BRIEF v.3.0 *UPGRADE*	£60.00
BRIEF OS/2 (U.K. Version)	£149.00
dBRIEF v.2.1 (U.K. Version)	£79.00
BRIEF with dBRIEF	£219.00
C-EDIT	£239.00
EMACS	£199.00
Epsilon	£110.00
ME Editor with C Source	£115.00
Micro ISPF	£110.00
Norton Editor	£49.00
PC/EDIT Plus	£179.00
Personal Rexx	£79.00
PI Editor	£119.00
SPF/PC	£149.00
Turbo Basic Editor Toolbox	£59.00
Vedit+	£115.00

HELP GUIDES AND TRAINING

dBASE on-line	£60.00
Norton Guides for ASM	£75.00
Norton Guides for BASIC	£75.00
Norton Guides for C	£60.00
Norton Guides for OS/2	£89.00
Norton Guides for Pascal	£75.00
Programmer's Library	£239.00

LINKERS & LIBRARIANS

ALINK *NEW*	£195.00
Archive Link	£239.00
Link & Locate ++	£245.00
LinkLok	£245.00
PLINK86 Plus *Now Polytron*	£495.00
PolyLibrarian I	£85.00
PolyLibrarian II	£125.00
.RTLink	£180.00

MAKE UTILITIES

PolyMake v.3.1	£125.00
MKS Make	£99.00
MKS Make 5 User	£320.00
MKS Make 10 User	£515.00

PROFILERS & OPTIMISERS

Codesifter	£75.00
DOSTRACE	£59.00
PC Metric	£120.00

PROTOTYPERS & GENERATORS

Automated Programmer	£605.00
Cogen	£275.00
COMPEDITOR	£110.00
Dan Bricklin Demo II	£125.00
Data Boss for C	£245.00
Data Boss for Pascal	£245.00
Genifer	£239.00
Instant Replay III	£89.00
Matrix Layout v.2.0	£129.00
Sycero db (U.K.)	£595.00
Sycero db Net (U.K.)	£745.00

SYNTAX ANALYSERS & LINTS

PC Lint v.3.0	£89.00
Pre-C	£150.00

TEXT & UNIX TOOLS

C Programmers Combination	£79.00
C ToolSet	£59.00
MKS LEX & YACC	£165.00
MKS Toolkit v.3.1 *NEW*	£149.00
MKS Toolkit v.3.1 *UPGRADE*	£50.00
PC Fortran Tools	£99.00
PC Yacc	£249.00
PolyAWK	£85.00
PolyAWK OS/2	£115.00
QPARSER Plus	£180.00
Text Management Utilities	£75.00
Turbo Professional	£75.00

TRANSLATORS

BAS_C Commercial	£229.00
Bastoc	£480.00
dBx dBASE to C v.4.7	£389.00
dBx Library Source	£CALL
dBx UNIX/XENIX Source	£CALL
FOR_C v.2.0 MS-DOS *NEW*	£565.00
FOR_C v.2.0 XENIX *NEW*	£630.00
FOR_C++ XENIX/386 *NEW*	£1195.00

VERSION CONTROL

M.K.S. R.C.S.	£125.00
M.K.S. R.C.S. 5 User	£485.00
M.K.S. R.C.S. 10 User	£905.00
PVCS Professional	£395.00
PVCS Network (5 User)	£1395.00
PVCS/PolyMake OS/2	£CALL

BACKUP - RECOVERY - REPAIR

dSalvage	£59.00
Fastback	£105.00
Fastback Plus	£115.00
Mace Utilities	£59.00
Norton Utilities Advanced	£89.00
PC Tools Deluxe	£69.00
XTree Net	£235.00

CACHE

PolyBoost II	£49.00

COMMS. & TERM. EMULATORS

Blast Communications	£149.00
Carbon Copy Express	£199.00
Carbon Copy Plus	£115.00
Crosstalk Mark 4	£145 00
Crosstalk XVI	£115.00
Mirror III	£75.00
PC Anywhere III	£85.00
Sidetalk II	£70.00
SmarTerm	£75.00
SmarTerm 125	£175.00
SmarTerm 220	£115.00
SmarTerm 2392	£135.00
SmarTerm 240	£199.00
SmarTerm 400	£99.00
SmarTerm 4014	£135.00
Z Stem PC-VT100	£59.00
Z Stem PC-VT220	£89.00
Z Stem PC-VT240	£175.00
Z Stem PC-VT4014	£59.00

DISK MANAGERS &OPTIMISERS

HTest/HFormat	£85.00
Mace 5	£65.00
Mace Gold	£95.00
Norton Utilities Advanced	£95.00
SpinRite II	£59.00

OP. SYSTEM ENHANCEMENTS

386MAX	£49.00
386MAX Professional	£80.00
DESQview	£79.00
Norton Commander	£55.00
Omni View	£49.00
PolyShell	£60.00
QDOS II	£45.00
Soft Bytes 386 Max	£45.00
Weiner Shell	£175.00
Power Bundle	£99.00
Windows/286	£59.00
Windows/386	£115.00

FAX OR TELEPHONE CREDIT CARD ORDERS VISA OR ACCESS

PRICING

All prices exclude V.A.T., but include first class recorded delivery.

EDUCATIONAL DISCOUNTS

Generous Educational discounts available on certain products.

CATALOGUE & FACT SHEETS

Call for our new Winter '90 catalogue, over 80 pages of Software Development Tools.

Personal Computer World May 1990

THE SOFTWARE CONSTRUCTION CO. LTD.

THE RED HOUSE, 84 HIGH ST., BUNTINGFORD, HERTS. SG9 9DJ (0763) 73455 FAX. (0763) 73460

CIRCLE NO. 897

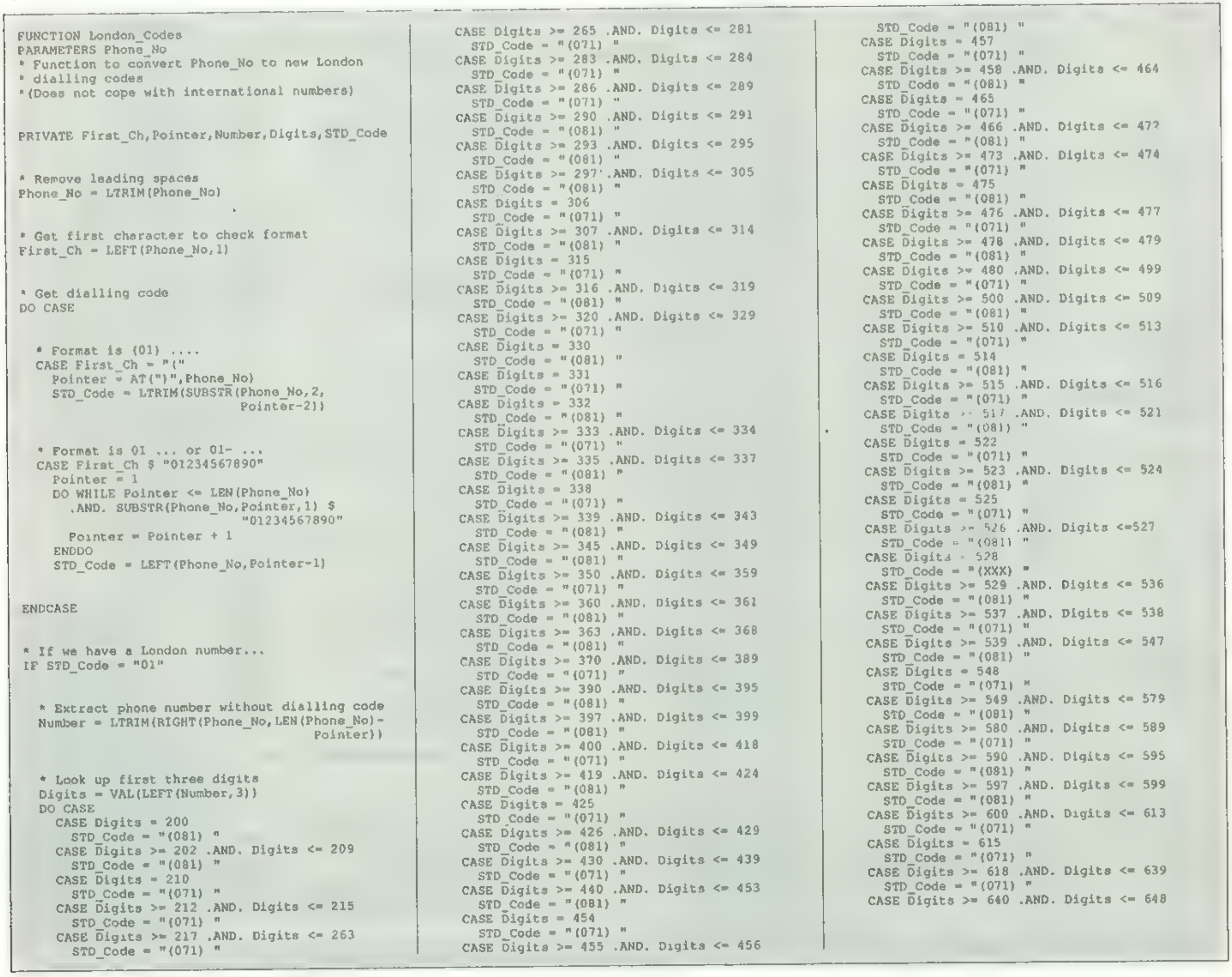
01 for dBASE

On the 6th of May, British Telecom is to abandon the London '01' dialling code, invalidating thousands of telephone databases. Dave Atkins has a dBASE program to put things to rights.

This month, British Telecom replaces London's 01 dialling code with two new codes: 071 for inner London and 081 for outer London (ie BT has added one extra decimal digit to the London phone number, but has only doubled the total of available numbers - digest and discuss). You will probably have seen some of the many adverts which have recently appeared for software utilities - usually ambitiously priced - which automate the major job of changing London telephone numbers. There is no need to spend money on such solutions. If you sit down and think about what needs to be done, the program to perform this process is straightforward. To save you this effort, this article presents my dBASE IV solution.

Storage

Telephone numbers are normally stored in character fields in dBASE; this allows the inclusion of formatting spaces, brackets and dashes, not to mention extension numbers. Some people use separate fields for the dialling code, the telephone number and any extension numbers; others combine them all in one field. My code deals with the case where the whole telephone number is combined in one field - adapting the program to other formats is a trivial exercise. Even within a single field, there are many ways in which telephone numbers can be entered - this is inevitably a punter-driven feature. For example, the .EXE Magazine number could be stored as (01) 994 6477, 01-994 6477, 01- 994 6477, (01)9946477 or 01-9946477, to name just a few. The important point is that the dialling



```
FUNCTION London_Codes
PARAMETERS Phone_No
* Function to convert Phone_No to new London
* dialling codes
*(Does not cope with international numbers)


PRIVATE First_Ch,Pointer,Number,Digits,STD_Code


* Remove leading spaces
Phone_No = LTRIM(Phone_No)


* Get first character to check format
First_Ch = LEFT(Phone_No,1)


* Get dialling code
DO CASE


  * Format is (01) ....
  CASE First_Ch = "("
    Pointer = AT(")",Phone_No)
    STD_Code = LTRIM(SUBSTR(Phone_No,2,
                            Pointer-2))


  * Format is 01 ... or 01- ...
  CASE First_Ch $ "01234567890"
    Pointer = 1
    DO WHILE Pointer <= LEN(Phone_No)
      .AND. SUBSTR(Phone_No,Pointer,1) $
                             "01234567890"
      Pointer = Pointer + 1
    ENDDO
    STD_Code = LEFT(Phone_No,Pointer-1)


ENDCASE


* If we have a London number...
IF STD_Code = "01"


  * Extract phone number without dialling code
  Number = LTRIM(RIGHT(Phone_No,LEN(Phone_No)-
                                     Pointer))


  * Look up first three digits
  Digits = VAL(LEFT(Number,3))
  DO CASE
    CASE Digits = 200
      STD_Code = "(081) "
    CASE Digits >= 202 .AND. Digits <= 209
      STD_Code = "(081) "
    CASE Digits = 210
      STD_Code = "(071) "
    CASE Digits >= 212 .AND. Digits <= 215
      STD_Code = "(071) "
    CASE Digits >= 217 .AND. Digits <= 263
      STD_Code = "(071) "
    CASE Digits >= 265 .AND. Digits <= 281
      STD_Code = "(071) "
    CASE Digits >= 283 .AND. Digits <= 284
      STD_Code = "(071) "
    CASE Digits >= 286 .AND. Digits <= 289
      STD_Code = "(071) "
    CASE Digits >= 290 .AND. Digits <= 291
      STD_Code = "(081) "
    CASE Digits >= 293 .AND. Digits <= 295
      STD_Code = "(081) "
    CASE Digits >= 297 .AND. Digits <= 305
      STD_Code = "(081) "
    CASE Digits = 306
      STD_Code = "(071) "
    CASE Digits >= 307 .AND. Digits <= 314
      STD_Code = "(081) "
    CASE Digits = 315
      STD_Code = "(071) "
    CASE Digits >= 316 .AND. Digits <= 319
      STD_Code = "(081) "
    CASE Digits >= 320 .AND. Digits <= 329
      STD_Code = "(071) "
    CASE Digits = 330
      STD_Code = "(081) "
    CASE Digits = 331
      STD_Code = "(071) "
    CASE Digits = 332
      STD_Code = "(081) "
    CASE Digits >= 333 .AND. Digits <= 334
      STD_Code = "(071) "
    CASE Digits >= 335 .AND. Digits <= 337
      STD_Code = "(081) "
    CASE Digits = 338
      STD_Code = "(071) "
    CASE Digits >= 339 .AND. Digits <= 343
      STD_Code = "(081) "
    CASE Digits >= 345 .AND. Digits <= 349
      STD_Code = "(081) "
    CASE Digits >= 350 .AND. Digits <= 359
      STD_Code = "(071) "
    CASE Digits >= 360 .AND. Digits <= 361
      STD_Code = "(081) "
    CASE Digits >= 363 .AND. Digits <= 368
      STD_Code = "(081) "
    CASE Digits >= 370 .AND. Digits <= 389
      STD_Code = "(071) "
    CASE Digits >= 390 .AND. Digits <= 395
      STD_Code = "(081) "
    CASE Digits >= 397 .AND. Digits <= 399
      STD_Code = "(081) "
    CASE Digits >= 400 .AND. Digits <= 418
      STD_Code = "(071) "
    CASE Digits >= 419 .AND. Digits <= 424
      STD_Code = "(081) "
    CASE Digits = 425
      STD_Code = "(071) "
    CASE Digits >= 426 .AND. Digits <= 429
      STD_Code = "(081) "
    CASE Digits >= 430 .AND. Digits <= 439
      STD_Code = "(071) "
    CASE Digits >= 440 .AND. Digits <= 453
      STD_Code = "(081) "
    CASE Digits = 454
      STD_Code = "(071) "
    CASE Digits >= 455 .AND. Digits <= 456
      STD_Code = "(081) "
    CASE Digits = 457
      STD_Code = "(071) "
    CASE Digits >= 458 .AND. Digits <= 464
      STD_Code = "(081) "
    CASE Digits = 465
      STD_Code = "(071) "
    CASE Digits >= 466 .AND. Digits <= 472
      STD_Code = "(081) "
    CASE Digits >= 473 .AND. Digits <= 474
      STD_Code = "(071) "
    CASE Digits = 475
      STD_Code = "(081) "
    CASE Digits >= 476 .AND. Digits <= 477
      STD_Code = "(071) "
    CASE Digits >= 478 .AND. Digits <= 479
      STD_Code = "(081) "
    CASE Digits >= 480 .AND. Digits <= 499
      STD_Code = "(071) "
    CASE Digits >= 500 .AND. Digits <= 509
      STD_Code = "(081) "
    CASE Digits >= 510 .AND. Digits <= 513
      STD_Code = "(071) "
    CASE Digits = 514
      STD_Code = "(081) "
    CASE Digits >= 515 .AND. Digits <= 516
      STD_Code = "(071) "
    CASE Digits >= 517 .AND. Digits <= 521
      STD_Code = "(081) "
    CASE Digits = 522
      STD_Code = "(071) "
    CASE Digits >= 523 .AND. Digits <= 524
      STD_Code = "(081) "
    CASE Digits = 525
      STD_Code = "(071) "
    CASE Digits >= 526 .AND. Digits <=527
      STD_Code = "(081) "
    CASE Digits = 528
      STD_Code = "(XXX) "
    CASE Digits >= 529 .AND. Digits <= 536
      STD_Code = "(081) "
    CASE Digits >= 537 .AND. Digits <= 538
      STD_Code = "(071) "
    CASE Digits >= 539 .AND. Digits <= 547
      STD_Code = "(081) "
    CASE Digits = 548
      STD_Code = "(071) "
    CASE Digits >= 549 .AND. Digits <= 579
      STD_Code = "(081) "
    CASE Digits >= 580 .AND. Digits <= 589
      STD_Code = "(071) "
    CASE Digits >= 590 .AND. Digits <= 595
      STD_Code = "(081) "
    CASE Digits >= 597 .AND. Digits <= 599
      STD_Code = "(081) "
    CASE Digits >= 600 .AND. Digits <= 613
      STD_Code = "(071) "
    CASE Digits = 615
      STD_Code = "(071) "
    CASE Digits >= 618 .AND. Digits <= 639
      STD_Code = "(071) "
    CASE Digits >= 640 .AND. Digits <= 648
```

Figure 1 - The dBASE conversion function

Figure 1 - continued

```
  STD_Code = "(081) "
CASE Digits >= 650 .AND. Digits <= 651
  STD_Code = "(081) "
CASE Digits >= 653 .AND. Digits <= 661
  STD_Code = "(081) "
CASE Digits >= 663 .AND. Digits <= 695
  STD_Code = "(081) "
CASE Digits = 696
  STD_Code = "(071) "
CASE Digits >= 697 .AND. Digits <= 699
  STD_Code = "(081) "
CASE Digits >= 700 .AND. Digits <= 704
  STD_Code = "(071) "
CASE Digits >= 706 .AND. Digits <= 716
  STD_Code = "(071) "
CASE Digits >= 718 .AND. Digits <= 739
  STD_Code = "(071) "
CASE Digits >= 740 .AND. Digits <= 752
  STD_Code = "(081) "
CASE Digits = 753
  STD_Code = "(071) "
CASE Digits >= 754 .AND. Digits <= 756
  STD_Code = "(081) "
CASE Digits = 757
  STD_Code = "(071) "
CASE Digits >= 758 .AND. Digits <= 761
  STD_Code = "(081) "
CASE Digits >= 763 .AND. Digits <= 764
  STD_Code = "(081) "
CASE Digits >= 766 .AND. Digits <= 771
  STD_Code = "(081) "
CASE Digits = 772
  STD_Code = "(071) "
CASE Digits = 773
  STD_Code = "(081) "
CASE Digits >= 774 .AND. Digits <= 775
  STD_Code = "(071) "
CASE Digits >= 776 .AND. Digits <= 778
  STD_Code = "(081) "
CASE Digits = 779
  STD_Code = "(071) "
CASE Digits >= 780 .AND. Digits <= 781
  STD_Code = "(081) "
CASE Digits = 782
  STD_Code = "(071) "
CASE Digits >= 783 .AND. Digits <= 789
  STD_Code = "(081) "
CASE Digits >= 790 .AND. Digits <= 796
  STD_Code = "(071) "
CASE Digits >= 798 .AND. Digits <= 799
  STD_Code = "(071) "
CASE Digits >= 800 .AND. Digits <= 809
  STD_Code = "(081) "
```

```
CASE Digits = 811
  STD_Code = "(081) "
CASE Digits >= 818 .AND. Digits <= 824
  STD_Code = "(071) "
CASE Digits = 826
  STD_Code = "(071) "
CASE Digits >= 828 .AND. Digits <= 829
  STD_Code = "(071) "
CASE Digits >= 831 .AND. Digits <= 839
  STD_Code = "(071) "
CASE Digits >= 840 .AND. Digits <= 859
  STD_Code = "(081) "
CASE Digits = 860
  STD_Code = "(071) "
CASE Digits >= 861 .AND. Digits <= 864
  STD_Code = "(081) "
CASE Digits = 865
  STD_Code = "(071) "
CASE Digits = 866
  STD_Code = "(081) "
CASE Digits = 867
  STD_Code = "(071) "
CASE Digits >= 868 .AND. Digits <= 871
  STD_Code = "(081) "
CASE Digits >= 872 .AND. Digits <= 873
  STD_Code = "(071) "
CASE Digits >= 874 .AND. Digits <= 879
  STD_Code = "(081) "
CASE Digits >= 881 .AND. Digits <= 886
  STD_Code = "(081) "
CASE Digits >= 888 .AND. Digits <= 894
  STD_Code = "(081) "
CASE Digits = 895
  STD_Code = "(XXX) "
CASE Digits >= 897 .AND. Digits <= 900
  STD_Code = "(081) "
CASE Digits = 901
  STD_Code = "(071) "
CASE Digits >= 902 .AND. Digits <= 909
  STD_Code = "(081) "
CASE Digits >= 911 .AND. Digits <= 912
  STD_Code = "(071) "
CASE Digits >= 913 .AND. Digits <= 914
  STD_Code = "(081) "
CASE Digits >= 915 .AND. Digits <= 918
  STD_Code = "(071) "
CASE Digits >= 920 .AND. Digits <= 925
  STD_Code = "(071) "
CASE Digits >= 927 .AND. Digits <= 939
  STD_Code = "(071) "
CASE Digits >= 940 .AND. Digits <= 944
```

code is entered first, and is separated from the rest of the number by a combination of brackets, dashes and spaces. We need to split the entry into two sections, so that we can process the dialling code and the remainder of the telephone number separately.

I first tackled this problem by writing a small program which used a separate indexed database file to store the new dialling codes. This worked fine, but I decided that a user defined function (UDF) would provide a more general solution to the problem, and had the advantage that it could be used just like any other dBASE function. `London_Codes`, which is the UDF routine shown in Figure 1, turned out to be quicker than my indexed database solution.

`London_Codes` splits into three logical sections. The first task is to extract the dialling code from the telephone number field. The process assumes that it is stored in one of the formats shown above. To determine the way in which the number has been stored, we check the first non-space character. There are two main possibilities: an

Figure 1 - continued

```
      STD_Code = "(081) "
    CASE Digits = 945
      STD_Code = "(XXX) "
    CASE Digits >= 946 .AND. Digits <= 954
      STD_Code = "(081) "
    CASE Digits >= 955 .AND. Digits <= 957
      STD_Code = "(071) "
    CASE Digits >= 958 .AND. Digits <= 961
      STD_Code = "(081) "
    CASE Digits = 962
      STD_Code = "(071) "
    CASE Digits >= 963 .AND. Digits <= 965
      STD_Code = "(081) "
    CASE Digits >= 967 .AND. Digits <= 969
      STD_Code = "(081) "
    CASE Digits >= 971 .AND. Digits <= 973
      STD_Code = "(071) "
    CASE Digits = 974
      STD_Code = "(081) "
    CASE Digits = 975
      STD_Code = "(XXX) "
    CASE Digits = 976
      STD_Code = "(071) "
    CASE Digits = 977
      STD_Code = "(081) "
    CASE Digits = 978
      STD_Code = "(071) "
    CASE Digits >= 979 .AND. Digits <= 981
      STD_Code = "(081) "
    CASE Digits = 982
      STD_Code = "(XXX) "
    CASE Digits >= 983 .AND. Digits <= 986
      STD_Code = "(081) "
    CASE Digits = 987
      STD_Code = "(071) "
    CASE Digits >= 988 .AND. Digits <= 989
      STD_Code = "(081) "
    CASE Digits >= 991 .AND. Digits <= 995
      STD_Code = "(081) "
    CASE Digits >= 997 .AND. Digits <= 998
      STD_Code = "(081) "
    OTHERWISE
      STD_Code = "(???) "
  ENDCASE

  Phone_No = STD_Code + Number


ENDIF


RETURN(Phone_no)
```

opening bracket (we must search for the corresponding closing bracket to find the dialling code) or a digit (we must search for either a space or a dash as the delimiting character). Once the format of the dialling code has been sorted out, the SUBSTR command can extract the dialling code.

If we find a London number (dialling code is 01), then we need to test the first three digits of the remainder of the number. The main part of the UDF is a CASE statement, which looks up the new dialling code. You will see that it includes some 'gaps', where there are no valid London telephone numbers. If you have any such non-valid telephone numbers in your database, these will be picked up, and the 01 dialling code replaced by '(???)'. There are also a few values where the number cannot be automatically converted - these entries will be replaced by '(XXX)'. It is certainly worth searching your database after translation, to see if either of these dialling codes is to be found. With a few manual alterations, your database will be up to date.

The final part of London_Codes simply returns the whole telephone number (dialling code plus the remainder of the telephone number), whether it has been changed or not. This result can be used by dBASE commands, such as REPLACE, to translate your databases:

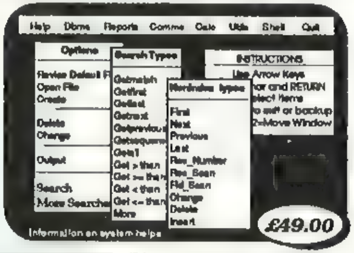
```
REPLACE ALL Phone
    WITH London_Codes(Phone)
```

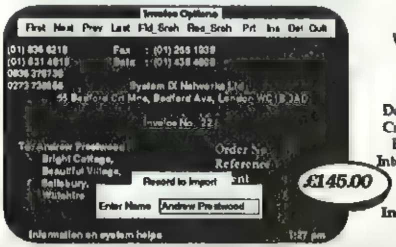
Remember that a translated London telephone number will contain extra characters (one extra digit, plus perhaps extra formatting), so do check, before you do anything, that the field is big enough to hold the revamped telephone number. Happy phoning!

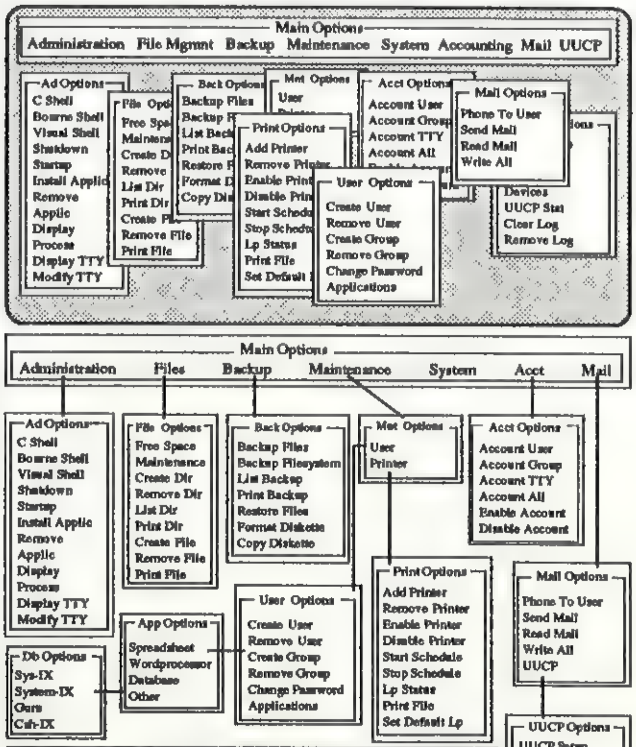
EXE

Dave Atkins works at the Computing Centre at Cardiff College, University of Wales, as a Microcomputer Support Officer. His job includes advising on the use of microcomputers, and providing courses on many microcomputer based packages.

The code in this article is available for downloading from the EXE conference on CIX. Alternatively, you can send a formatted diskette and an SAE to the editorial address (given on page 2) and we will copy it. Mark your envelope '01 for dBASE'.


Help Dbms Reports Comms Calc Utils Shell Quit
Options
Search Types
INSTRUCTIONS
Use Arrow Keys
Information on system helps

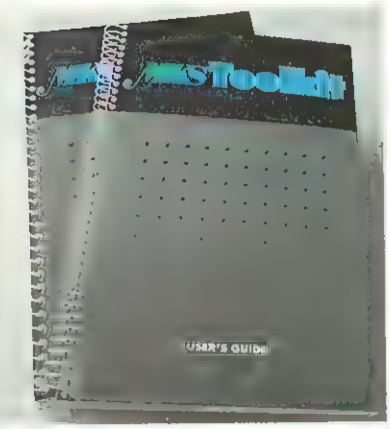




"The MKS Toolkit is an amazingly faithful replication of a System V UNIX™ environment."

- UNIX Review

Essential UNIX Tools for any PC user



"The entire MKS documentation package proved excellent in every respect"

Daniel McAuliffe, IEEE Computer, Jan. 1990

We can tell you that MKS Toolkit offers both experts and novices the purest form of UNIX utilities in a DOS or OS/2 environment. Fortunately, we don't have to. Software reviewers, universities and major corporations all over the world are discovering MKS Software and how it can help their programmers and software developers make the most of their creative talents.

With our proprietary code, MKS Toolkit offers you more than 150 UNIX System V. Release 4 compatible tools for DOS or OS/2. With MKS Toolkit, your computer or clone becomes a comfortable environment for shells, string matching, editing, file manipulation, and more.

In this one package you will receive a complete:

- MKS Vi full – screen editor
- MKS Korn Shell – user programmable interface
- MKS AWK – programming and prototyping language
- profiler – to analyse the execution time of code
- egrep/fgrep – fast pattern matching
- archivers, sort/merge, encryption, data compression & much more

Over 150 commands that help you get the most productivity out of your PC.

Easy Installation

With MKS's menu-driven installation you can be up and running in less than 10 minutes, no matter what your level of computer expertise.

"MKS software is absolutely the best in its class. Don't mix environments without it."

Grover Righter, Novell

Interconnectivity & Lan Licences

MKS Toolkit interacts well on most standard PC and PS/2 networks. Combined with Novell Netware™, the most popular LAN for PC's, MKS Toolkit creates a UNIX time sharing system in DOS or OS/2 organizations. UNIX workgroups can now hook up all of their PCs using PC-NFS™ and MKS Toolkit, enabling you to use a PC as a UNIX workstation and off-load your mini or main machine.

Spanning Novice to Expert

This versatile software reflects it's users' needs. UNIX experts will have all the familiar power they demand while novices will find our documentation filled with excellent tutorial material.

Organizations such as AT&T, Hewlett-Packard, ITT, and NCR – all heavily committed to the UNIX system – use MKS Toolkit to create a standard operating environment. Universities from King's College to the University of Berlin use MKS Toolkit to enrich personal research computing environments and double the potential of their PC teaching labs.

"With this package, you can become familiar with the UNIX environment on your microcomputer, with DOS only a keystroke away."

Byte Magazine, May 1989

Cost-Effective Learning Tool

If your organization is committed to moving into the UNIX environment, then MKS Toolkit is the perfect learning path. DOS or OS/2 users retain the familiar world of their PC keyboard and programs yet can move to a desktop UNIX environment at a keystroke. Exposure to new commands and functionality now becomes an integral part of the novice's working day.

MKS Programming Platform

MKS Toolkit can be purchased as an individual product or you can buy MKS Programming Platform which contains MKS Toolkit and three other MKS products. MKS Platform helps smooth out the details of programming and software development by adding time-saving utilities such as: MKS RCS (Revision Control system), MKS Make (an automated program builder), MKS LEX & YACC (compiler learning and construction tools).

In all, you simply cannot find a more complete set of commands and utilities to get from DOS or OS/2 to UNIX. Once you have used MKS Toolkit, programming will never be the same again.

US List Price

MKS Toolkit (DOS & OS/2)	$399.00
MKS Toolkit (DOS/2 only)	$349.00
MKS Toolkit (DOS only)	$249.00

Please contact MKS dealers for UK domestic prices.

The Software Construction Co. Ltd (0763) 73455; fax (0763) 73460
Grey Matter (0364) 53499
Systems Science (01) 833-1022; fax (01) 837-6411

MKS

35 King Street North, Waterloo, Ontario, Canada N2J 2W9

Put The Power Of UNIX In Your PC.

MKS, MKS Make, MKS Toolkit Korn Shell, and MS RCS are trademarks of Mortice Kern Systems Inc. UNIX is a trademark of AT&T
PC-NFS is a trademark of Sun Microsystems

CIRCLE NO. 900

A new way through the barrier

To get at the full address space of the 80386 processor, you must run it in protected mode, says Conventional Wisdom. But David Bailey knows about some undocumented features...

Until people stop running the MS-DOS operating system on 386 and 486 machines - and there is little sign of this happening at the moment - there will remain a need to address more than 640 KB of RAM from 'real mode', the mode in which the 80386 chip is emulating its 8086 ancestors. Programs executing in real mode are limited to a 1 MB address space because addresses are calculated from the 16-bit segment and offset values by the formula:

Absolute address = (16 * segment) + offset

Using this formula, it is possible to generate addresses up to 10FFEFh - slightly more than 1 MB - but the 8086 truncated the address to 20 bits, as we shall see. In practice, the memory between 640 KB and 1 MB is reserved for screen memory and the BIOS ROM. The remaining 640 KB of RAM must contain a copy of MS-DOS, plus any TSR programs you may be running. The practical limit on program size is about 400-500 KB, depending on the version of MS-DOS you are using and the size of any TSRs that you have loaded.

Two approaches are commonly used to bypass the 640 KB limit from MS-DOS. The first uses the standard interface to expanded memory (LIM) to switch blocks of memory into an unused portion of the address space above 640 KB. Alternatively, on 386 machines, this can be achieved by running MS-DOS in virtual mode, rather than real mode. In virtual mode the processor can make use of its virtual memory hardware to switch 4 KB pages of memory into the bottom megabyte of address space using a 386 memory manager.

Block switching is, of course, a very inelegant way of accessing memory - especially from high-level languages - and a variety of so called 'DOS extenders' now offer a second, much more radical solution to the problem. At Salford University we use a home-made DOS extender, called DBOS. We use it to run our FORTRAN 77 compiler, FTN77, under MS-DOS, with the processor in 32-bit protected mode (please see 'Anatomy of a DOS Extender', .EXE, July 1989 for further details). The DOS extender works by switching to protected mode to execute the program, and returning to real mode to access DOS or BIOS services. This technique lets you execute multi-megabyte programs from MS-DOS, and it has significantly reduced the pressure for a 'true' 32-bit operating system.

It is generally thought to be inherently impossible to address all the memory available

However, there are still circumstances when it would be convenient to address all the extra memory available on a modern PC from real mode. This is generally thought to be inherently impossible - you won't find it covered in any of the reference books. But recently I stumbled across an undocumented way of doing it. I rang Intel, to check the validity of my findings. Not only is the technique described below known about, it is also supported, so it should work on such chips as the 386-SX, the 486 and, in due course, the 586.

How it works

In order to explain this trick it is necessary to review the way in which the 80386 extends the 8086 architecture. Each of the 16-bit registers (AX, BX, CX, DX, SI, DI, BP, IP, and SP) is extended to 32-bits (EAX, EBX etc). These 32-bit registers are accessible from real mode using an 'operand size' prefix byte of 66h. MASM 5.0 'knows' about these registers, and will plant the prefix byte if you specify a 32-bit operation in 16-bit code. Using an 'address size' prefix (67h) it is also possible to make use of a whole new set of address modes which, at first sight, appears to let you address 4 GB of memory. A typical instruction, which would require both prefixes to run in real mode, would be:

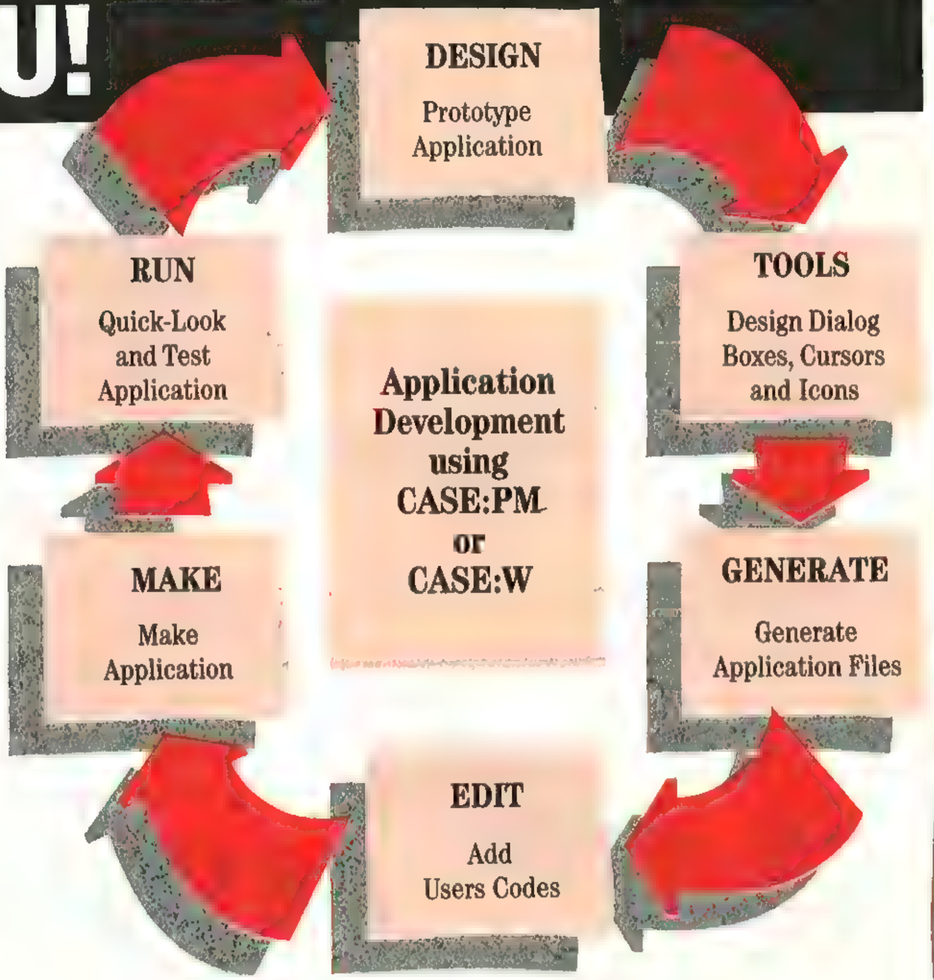
```
MOV     EAX,DS:[ECX]
```

This would use the contents of ECX as an offset into the DS segment to produce an absolute address which could span 4 GB. You can code this instruction with MASM 5.0 and execute it in real mode. However, if the contents of ECX exceed 64 KB, the 80386 traps this instruction with a 'General Protection Exception' (interrupt 0Dh). The problem is that in real mode the size of each segment is set to 64 KB. The check to prevent addressing outside a segment is really part of the protection mechanism in protected mode, and the limit of 64 KB is set up when the chip is reset.

This 64 KB limit is almost invisible to real mode programmers. It manifests itself when you try to use 32-bit addressing using the address-size prefix as above, or when

CASE:PM or CASE:W

THE TOOLS THAT WRITE PM OR WINDOWS CODE FOR YOU!



DESIGN
Prototype Application

TOOLS
Design Dialog Boxes, Cursors and Icons

GENERATE
Generate Application Files

EDIT
Add Users Codes

MAKE
Make Application

RUN
Quick-Look and Test Application

Application Development using CASE:PM or CASE:W

Writing applications for Windows or OS/2 Presentation Manager graphical user interface just became a lot easier, because now there's CASE:PM (for Presentation Manager) and CASE:W (for Windows).

CASE: PM™

Both are application prototypers and code generators that write complete C skeleton programs for you based on a description of the 'user interface'.

APPLICATION PROTOTYPING

Just by 'pointing and clicking' at various parts of a window you can now create a Windows or PM user interface – in minutes!

APPLICATION GENERATION

Once you have the 'look and feel' of the application, CASE:PM or CASE:W produces C source code, resource scripts, definition files and make files... *automatically.*

Run *MAKE* and the files are compiled and linked producing a standalone executable with no run-time environments or royalties to pay.

CASE: W™

APPLICATION MAINTENANCE

Once you have the skeleton you add the application specific code. The generated source code is fully documented, with instruction as to where to put the application specific lines, leaving you free to concentrate on the application itself.

Change the user interface design and CASE:PM or CASE:W will generate the source files with the new user interface – but retaining the code you've added.

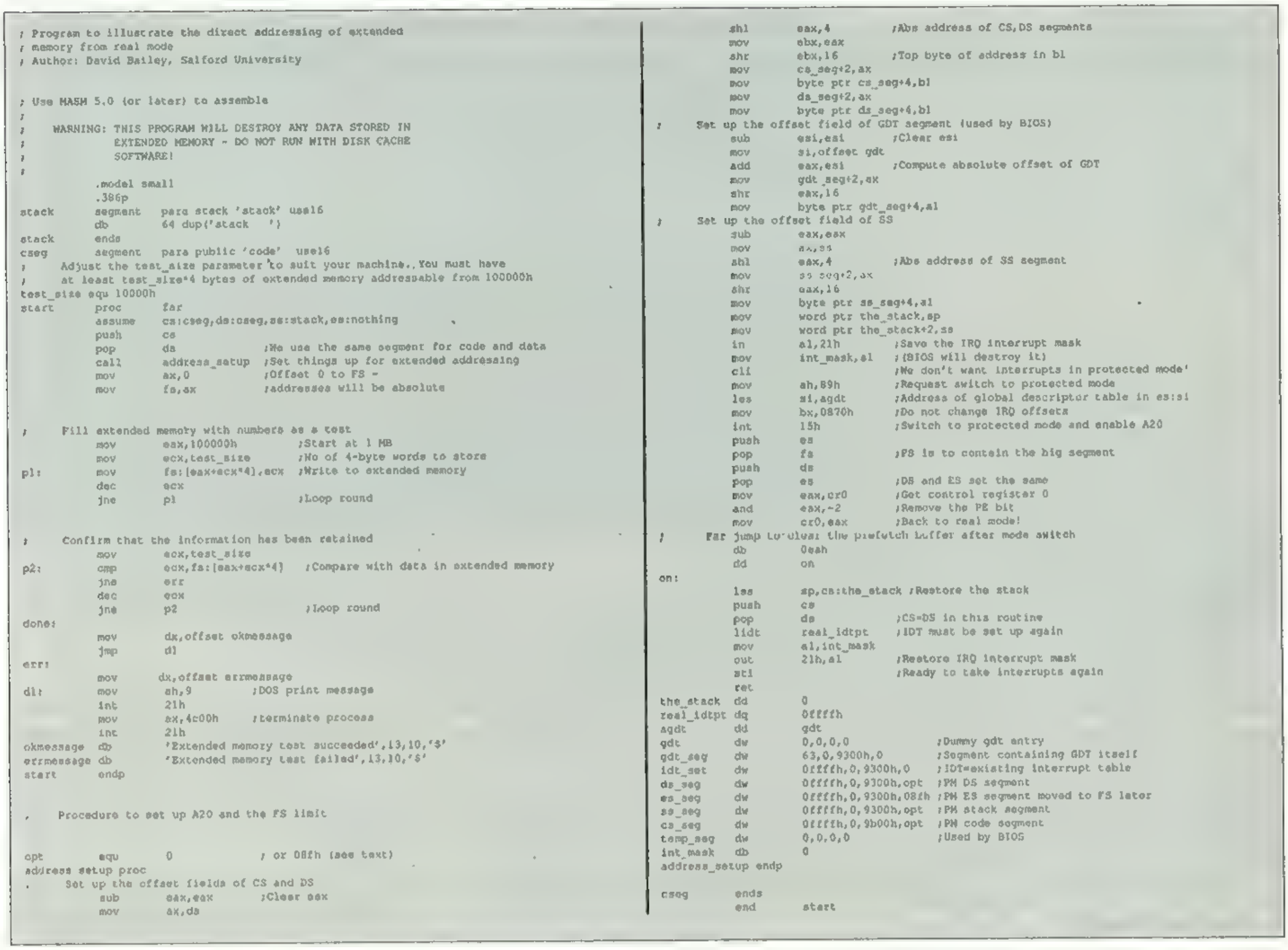
CASE:PM or CASE:W will reduce your learning curve and increase your productivity. For information and technical specifications call the Products Group on 0285 655888 FAX (0285 650537), or write to us.

QA

Cecily Hill Castle
Cirencester, Gloucester GL7 2EF
0285 655888

CIRCLE NO. 901

Figure 1 - Program to access 32-bit address space in real mode



```
; Program to illustrate the direct addressing of extended
; memory from real mode
; Author: David Bailey, Salford University


; Use MASM 5.0 (or later) to assemble
;
;    WARNING: THIS PROGRAM WILL DESTROY ANY DATA STORED IN
;             EXTENDED MEMORY - DO NOT RUN WITH DISK CACHE
;             SOFTWARE!
;
          .model small
          .386p
stack     segment   para stack 'stack' use16
          db        64 dup('stack   ')
stack     ends
cseg      segment   para public 'code'  use16
;     Adjust the test_size parameter to suit your machine., You must have
;     at least test_size*4 bytes of extended memory addressable from 100000h
test_size equ 10000h
start     proc      far
          assume    cs:cseg,ds:cseg,ss:stack,es:nothing
          push      cs
          pop       ds              ;We use the same segment for code and data
          call      address_setup   ;Set things up for extended addressing
          mov       ax,0            ;Offset 0 to FS -
          mov       fs,ax           ;addresses will be absolute


;     Fill extended memory with numbers as a test
          mov       eax,100000h          ;Start at 1 MB
          mov       ecx,test_size        ;No of 4-byte words to store
p1:       mov       fs:[eax+ecx*4],ecx   ;Write to extended memory
          dec       ecx
          jne       p1                   ;Loop round


;     Confirm that the information has been retained
          mov       ecx,test_size
p2:       cmp       ecx,fs:[eax+ecx*4]   ;Compare with data in extended memory
          jne       err
          dec       ecx
          jne       p2                   ;Loop round
done:
          mov       dx,offset okmessage
          jmp       d1
err:
          mov      dx,offset errmessage
d1:       mov       ah,9          ;DOS print message
          int       21h
          mov       ax,4c00h      ;terminate process
          int       21h
okmessage db        'Extended memory test succeeded',13,10,'$'
errmessage db       'Extended memory test failed',13,10,'$'
start     endp


,    Procedure to set up A20 and the FS limit


opt       equ       0               ; or 08fh (see text)
address setup proc
.     Set up the offset fields of CS and DS
          sub       eax,eax       ;Clear eax
          mov       ax,ds
          shl       eax,4         ;Abs address of CS,DS segments
          mov       ebx,eax
          shr       ebx,16        ;Top byte of address in bl
          mov       cs_seg+2,ax
          mov       byte ptr cs_seg+4,bl
          mov       ds_seg+2,ax
          mov       byte ptr ds_seg+4,bl
;     Set up the offset field of GDT segment (used by BIOS)
          sub       esi,esi       ;Clear esi
          mov       si,offset gdt
          add       eax,esi       ;Compute absolute offset of GDT
          mov       gdt_seg+2,ax
          shr       eax,16
          mov       byte ptr gdt_seg+4,al
;     Set up the offset field of SS
          sub       eax,eax
          mov       ax,ss
          shl       eax,4         ;Abs address of SS segment
          mov       ss seg+2,ax
          shr       eax,16
          mov       byte ptr ss_seg+4,al
          mov       word ptr the_stack,sp
          mov       word ptr the_stack+2,ss
          in        al,21h        ;Save the IRQ interrupt mask
          mov       int_mask,al   ;(BIOS will destroy it)
          cli                     ;We don't want interrupts in protected mode'
          mov       ah,89h        ;Request switch to protected mode
          les       si,agdt       ;Address of global descriptor table in es:si
          mov       bx,0870h      ;Do not change IRQ offsets
          int       15h           ;Switch to protected mode and enable A20
          push      es
          pop       fs            ;FS is to contain the big segment
          push      ds
          pop       es            ;DS and ES set the same
          mov       eax,cr0       ;Get control register 0
          and       eax,-2        ;Remove the PE bit
          mov       cr0,eax       ;Back to real mode!
;      Far jump to clear the prefetch buffer after mode switch
          db        0eah
          dd        on
on:
          lss       sp,cs:the_stack ;Restore the stack
          push      cs
          pop       ds            ;CS=DS in this routine
          lidt      real_idtpt    ;IDT must be set up again
          mov       al,int_mask
          out       21h,al        ;Restore IRQ interrupt mask
          sti                     ;Ready to take interrupts again
          ret
the_stack dd        0
real_idtpt dq       0ffffh
agdt      dd        gdt
gdt       dw        0,0,0,0              ;Dummy gdt entry
gdt_seg   dw        63,0,9300h,0         ;Segment containing GDT itself
idt_set   dw        0ffffh,0,9300h,0     ;IDT=existing interrupt table
ds_seg    dw        0ffffh,0,9300h,opt   ;PM DS segment
es_seg    dw        0ffffh,0,9300h,08fh  ;PM ES segment moved to FS later
ss_seg    dw        0ffffh,0,9300h,opt   ;PM stack segment
cs_seg    dw        0ffffh,0,9b00h,opt   ;PM code segment
temp_seg  dw        0,0,0,0              ;Used by BIOS
int_mask  db        0
address_setup endp

cseg      ends
          end       start
```

you try to reference a memory operand which spills over the top of a segment. For example:

```
MOV    SI,0FFFFh
MOV    AX,[SI]
```

The second instruction will generate interrupt 0Dh when executed on an 80386/486. The 64 KB limit also shows up if you execute instructions off the top of your code segment.

In protected mode, segments can have sizes ranging from 1 byte to 4 GB, depending on information contained in the Global Descriptor Table (GDT). Each time a segment register is loaded in protected mode, the segment register contents are used to index the GDT to extract the size, offset and certain other information about the segment. If the processor is returned to real mode, by resetting the PE bit in CR0, the current values are frozen. Subsequently, only the offset information is altered when a segment register is loaded, and the GDT is not referenced. By the way, it is this on-chip information which can be loaded directly by the LOADALL undocumented instruction ('The Code Page', .EXE, September 1989). However, this instruction is only guaranteed to work on 286 machines. It is also extremely clumsy to use, as it takes its data from a fixed area of memory normally used by MS-DOS.

DOS extenders like DBOS, which must repeatedly switch between real and protected modes, set up the segments with appropriate values before the switch back to real mode. These values then become frozen in, as described above. The *80386 Programmer's Reference Manual* advises that all segment registers should be set up with 64 KB limits for real mode, without saying what happens if you use other values. In fact, you can set up the FS and GS segments as you wish - since normal real mode makes no use of these registers, which do not exist on pre-80386 CPUs.

Now you can see how to address extended memory from real mode. You make one brief excursion into protected mode, and set the FS segment to a size of 4 GB. You then return to real mode, freezing in the new size information. Now, by using an FS-prefixed instruction, you can address *anywhere* in memory! You must take care of one more detail. As mentioned above, the 80386 does not accurately emulate an 8086 in real mode in one respect: it does not truncate addresses at 1 MB. This means that, for example, the address FFFF:1000 refers to a location above 1 MB on a 386 machine and a location near the bottom of memory on an 8086 machine. Unfortunately, some early MS-DOS software (dismally) relies on this address wraparound, and 80386 PC manufacturers have incorporated special hardware to emulate this behaviour. This hardware disables address line 20 (A20), and is known as the gate A20 switch.

We can use a BIOS call to switch us into protected mode and enable gate A20 in one operation. I have found this function too slow when performing frequent switches, as in my DOS extender, but for the purposes of this article, the switch need only be done once, so efficiency is not an issue. The BIOS provides no way back from protected mode - indeed it provides no services at all in protected mode - so the return to real mode is achieved explicitly by resetting the PE bit in CR0. This leaves gate A20 enabled. The code required to perform all this is shown in Figure 1. The full explanation for the values in the GDT can be found in the *80386 Programmer's Reference manual*. Since only very old software needs gate A20 disabled, you can probably leave it enabled after your program returns to DOS. Note that, back in real mode, FS is explicitly loaded with zero before use. This will force the offset of the FS segment to zero, and will mean that the address will be absolute. There is, of course, no need for a segment address component offset with a 32-bit segment size.

The ability to access extended memory directly has obvious value to the writers of TSR utilities. Normally, these programs must use memory sparingly since, once loaded, they permanently reduce the memory available below 640 KB. Once the FS segment and gate A20 have been set, they are unlikely to be reset. Only software which uses a LIM emulator or DOS extender might disturb these values. (My own program, DBOS, from version 2.01, explicitly leaves the machine set up ready to address extended memory). If in doubt, it is possible to check that all is well before attempting to access extended memory. Simply hook interrupt 0Dh (the general protection interrupt) and execute an instruction to address above 1 MB. If the program takes the interrupt, you know that you need to return to protected mode to set things up again. This test should be performed with external interrupts disabled, since external hardware, such as network cards, may use the interrupt. To test gate A20, simply compare a few bytes at location 0 with the corresponding values just above 1 MB. If they agree, gate A20 has presumably been disabled and you should re-execute the address_setup procedure of Figure 1.

Code as well

Now that we can address unlimited memory from real mode, you may be wondering if it is possible to execute code from extended memory while in real mode. The answer is yes - but at a price. First, you must extend the size of the other segments. This can be done by changing the opt symbol in Figure 1 to 08fh. Next, it is necessary to move some code into extended memory, and transfer control to it with a 32-bit far return or jump. Control can be returned beneath 1 MB by executing a 16-bit far return. Figure 2 shows some example code that I have executed in this way, using the modified form of the procedure address_setup to set up the CPU. I ran the program under DOS's DEBUG, to pick up the embedded interrupt 3 call. DEBUG successfully reported CX containing 1234h - proving that the code in extended memory had executed correctly.

Figure 2 - Sample program code, running in 32-bit address space

```
      call   address_setup ;Set things up for extended addressing
      mov    ax,0       ;Offset 0 to FS - addresses will be absolute
      mov    fs,ax
      cli          ;Interrupts off while in extended memory
;    Move the instructions
      mov    eax,200000h
      mov    si,offset codex
      mov    edx,[si]
      mov    fs:[eax],edx
      mov    edx,[si+4]
      mov    fs:[eax+4],edx
;    Leave a 16-bit far return on the stack to get back
      call   far ptr tl
      int    3        ;Display registers with DEBUG
      mov   ax,4c00h ;terminate process
      int   21h
tl:
      push   0        ;Push a CS of 0 extended to 32 bits
      push   0        ;
      mov    eax,200000h   ;Desired EIP
      push   eax
      db    66h       ;Operand size prefix
      retf           ;Goto 0:200000
;
;    The following two instructions are moved to 0:200000
;    and executed there
codex:
      mov    cx,1234h    ;Leave a signature in the registers
      retf           ;Return to the int 3 instruction
```

Every JMP, CALL, RET or conditional jump executed in extended memory must be 32-bit (ie have the operand size prefix attached). Interrupts must be disabled while executing code in extended memory, since an interrupt would only save IP, not EIP. If you wish to transfer your stack into extended memory, while executing instructions up there, you must make a further modification to the real mode environment. The last word of the stack segment entry in the GDT of Figure 1 must be changed from 'opt' to 0cfh. This will cause all stack references to operate using ESP rather than SP. Before switching to protected mode, it is advisable to clear the top half of ESP if you are experimenting with big stacks.

You can execute code from extended memory while running in real mode - but the price is very high

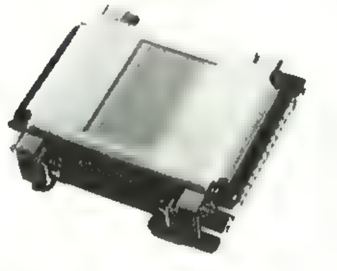
I am not sure whether it is worth manipulating code - as opposed to data - above 1 MB. It would clearly be necessary to descend from time to time to take external interrupts or to issue interrupts to DOS or BIOS. However, my tests show that MS-DOS still seems to run with 4 GB segments and a big stack.

Please remember that none of the techniques described here will work in virtual mode. That means that you could not use them in conjunction with most memory managers, or with software such as DESQview/386.

Finally, you may be wondering what use this technique has been to us. Programs compiled with our FORTRAN compiler (including the compiler itself) execute in 32-bit protected mode using DBOS and have no need for this trick in order to address extended memory. However, we have used the technique to implement a disk cache, based on any extended memory which DBOS is not using. Memory can be released from the disk cache when it is needed, and returned to the cache after program termination. This is an extremely efficient use of extended memory, since it is not necessary to partition extended memory into a disk cache and program area. The protection mechanisms ensure that an errant program cannot corrupt the contents of the disk cache. The cache operates by hooking the disk interrupt (13h) and addressing extended memory as described. The drawback is that the disk cache is not available when DBOS is run under DESQview/386, for the reasons explained above. We have no plans to execute serious real mode code in extended memory. EXE

David Bailey is part of the group at Salford University which developed the FTN77/386 compiler. The compiler was reviewed in the May '89 issue of .EXE Magazine.

SOFTLoK^tm & SOFTLoK PLUS^tm software protection systems



- For IBM PC, PS/2 and compatibles
- Installed in seconds in the parallel printer port
- Assembler & high level source code provided
- Fast response
- Transparent to printer
- Small physical size

SOFTLoK

SOFTLoK units contain a code that can be read and verified by routines provided in assembler and a variety of high level languages.

Price: 10-99 units £12.00, 100+ units £8.00
Evaluation kit £20.00

SOFTLoK PLUS

SOFTLoK PLUS units contain a small area of non-volatile memory protected from unauthorised reading or writing by a password. Both the memory area and the password can be changed at any time by the routines provided in assembler & several high level languages. As the memory can even be altered in the field it can contain counters & expiry dates as well as serial numbers etc.

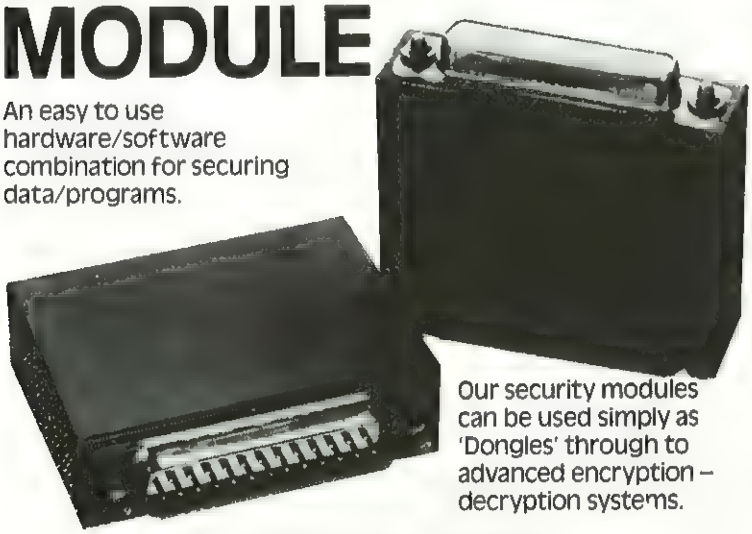
Price: 10-49 units £40.00, 50+ units £35.00
Evaluation kit £40.00

SOFTLoK International Limited
Tel: (0254) 772220 Fax: (0254) 705956

CIRCLE NO. 903

SOFTWARE SECURITY MODULE

An easy to use hardware/software combination for securing data/programs.



Our security modules can be used simply as 'Dongles' through to advanced encryption – decryption systems.

- ★ For IBM PC family (and compatibles)
- ★ Variants for other systems
- ★ Advanced construction – rugged and compact
- ★ Proven throughout the world
- ★ Transparent operation – user friendly
- ★ Driver software supplied for variety of applications

CTL

Control Telemetry of London
11 Canfield Place, London NW6 3BT
Tel: 071 -328 115 Fax: 071 -328 9149

CIRCLE NO. 904

System Science

C COMPILERS

Microsoft C ver 6.0	£245.00	Turbo C	£75.00
Microsoft Quick C2	£65.00	Turbo C professional	£155.00
JPI Topspeed C-std	£149.00	Aztec C86 Comm	£245.00
JPI Topspeed C-ext	£295.00	Lattice C (new ver 6.0)	£185.00
Zortech C+ + Ver 2.0	£125.00	Glockenspiel Common View	£375.00
Zortech C+ + Prof Der	£275.00	Quick C with Quick ASM	£129.00

C GRAPHICS

Halo-88 (also Qb, Fort, Pas)	£195.00	Essential Graphics	£225.00
Metawindows-no royalties	£185.00	Metawindows Plus	£245.00
Ingraf-C-source, no royalties	£175.00	PCX effects	£85.00
PCX programmers Toolkit	£165.00	MS Windows SDK	£295.00

C DATAFILE

Btrieve (also QB, Pas, Fort etc)	£159.00	Ctree — source	£255.00
Btrieve/N -network	£395.00	Fabs Plus	£125.00
XQL addon for Btrieve	£525.00	Code BASE 4	£195.00
Xtrieve +	£395.00	CQL for Ctree or Btrieve	£295.00

C SCREENS

Greenleaf Data Windows	£195.00	Aspen Cubes — source	£225.00
Panel Plus II	£275.00	Formation Source	£265.00
C Worthy	£call	Cscape with Look & Feel	£315.00

C COMMS AND OTHERS

Greenleaf Comms	£175.00	Essential Comms s'rce	£195.00
Blaise C Asynch Manager	£125.00	C Lisp — source	£295.00
Blaise Turbo C Tools- new ver	£95.00	Blaise C Tools Plus	£95.00
Greenleaf Functions	£125.00	Greenleaf Super Funcs	£165.00

PROGRAMMERS TOOLS

Microsoft Assembler 5.1	£90.00	Turbo Assembler/Debugger	£99.00
Brief Editor (DOS or OS/2)	£150.00	Epsilon Editor (DOS/OS/2)	£145.00
VS Designer (case)	£375.00	Vedit Plus new ver	£99.00
Plink 86 Plus	£395.00	RT Link Plus	£375.00
PolyMake	£115.00	PVCS Corporate	£375.00
MKS Toolkit	£155.00	Vq matchmaker & editor	£175.00

CROSS DEVELOPMENT

2500AD Cross Assemblers	£135.00	2500AD Simulators	£115.00
Avocet Cross ASM	£295.00	Avocet Simulators	£395.00
IAR Cross Assemblers	£185.00	Avocet C Cross	£695.00
IAR C Cross Z80, 6301 etc	£995.00	Introl C Cr. 6809, 6301, 68000	£1445.00
Aztec C Cross	£275.00	LMI Forth Metacompiler	£495.00

FORTRAN ETC

Microsoft Fortran with CodeView	£245.00	RM Fortran with Forte	£465.00
Lahey F77L (new version)	£395.00	Halo 88	£195.00
Lahey F77/32EM for 386s (new version)	£795.00	Ingraf Fortran source	£175.00
Salford FTN77-386	£695.00	Enhanced Graphics T'kit s'rce	£175.00
GSS GKS Graphics	£385.00	FortLib source	£95.00

PASCAL ETC

Turbo Pascal 5.5	£75.00	Blaise Power Tools Plus	£95.00
Turbo Pascal Professional	£155.00	Blaise Turbo Asynch	£115.00
Quick Pascal	£65.00	Microsoft Pascal 4.0	£165.00
Fabs Plus Datafile	£125.00	Many more for Turbo Pascal	£call

BASIC ETC

Microsoft QuickBasic ver 4.5	£65.00	Quick Comm	£95.00
Microsoft Basic 7 (new)	£275.00	Crecent Quick Pak Prof	£115.00
Btrieve	£159.00	Fabs Plus Datafile	£125.00
Blaise Power Screen	£95.00	Many more	£call

OTHER LANGUAGES

JPI Modula-2	£59.00	Logitech Modula-2 Develop	£175.00
MuLisp 87 Interpreter	£175.00	MuLisp-87 Compiler & interp	£225.00
Smalltalk V	£65.00	Smalltalk 286	£129.00
PC Forth by LMI	£95.00	UR/Forth by LMI	£225.00
Microsoft Cobol II	£445.00	PC Logo	£75.00

UTILITIES & OTHERS

Uniform/PC read CP/M disks	£62.00	Dan Bricklins Demo II	£175.00
Spin Rite II	£69.00	Page Garden	£85.00
Unidos with 8MHz Z80	£125.00	Sourcer with BIOS	£115.00
Q Ram (new)	£55.00	Vfeature Deluxe	£85.00
MS Mouse (Bus/Ser)	£95.00	Clarion Professional	£475.00
Above disc Lim Dr	£69.00	Math CAD 2.5	£345.00
Microstat II	£295.00	Derive — computer Algebra	£115.00

386 AND OS/2

385Max Prof	£95.00	VM/386	£145.00
DesqView 386 (new version)	£125.00	Xenix 386 — new price list	£call
OS/2 1 2 Stand Ed (IBM)	£call	OS/2 PM Toolskit	£295.00
Many OS/2 Libs & utils	£call	Many 386 Utils	£call
QEMM (new)	£60.00	Win Pro/PM	£250.00
Smalltalk/PM	£295.00	Multiscope Debugger	£195.00

All prices are exclusive of VAT but do include UK delivery.

A sample only of our extensive range. Please enquire for unlisted items

Access

Prices are subject
to change —
Please confirm

3-5 Cynthia Street,
London N1 9JF
Tel: (01) 833-1022,
Fax: (01) 837-6411

BARCLAYCARD
VISA

CIRCLE NO. 905

Opening Windows

Resources form an important element of Microsoft Windows programming. The Whitewater Resource Toolkit aims to simplify creating and maintaining a program's resources. Dave Jewell checks it out.

Resource: means of supplying a want; stock that can be drawn on.

Under Windows, an application program's various menus, dialog boxes, icons, bitmaps, and so on are generally packaged as 'resources'. These are chunks of binary data, produced by a special resource compiler, then appended to the Windows .EXE file after the program itself has been compiled and linked. All the tools required to carry out this process are to be found in the Windows SDK package. Unfortunately, when it comes to getting the appearance of a dialog box or an icon just right, the process of refinement seems endless. Each little fix forces you to rebuild your program, run Windows, exit Windows, edit the dialog box definition, rebuild the program, and so on ad infinitum.

The Whitewater Group produces the Whitewater Resource Toolkit, an interactive resource editor designed to take the frustration out of Windows resource development. This is the same company that developed the object-oriented Actor programming environment, and, in fact, the Resource Toolkit was itself written in the Actor language.

The biggest advantage of using the Toolkit is that you can create, edit and display resources without having to leave the Windows environment. This saves an immense amount of time, particularly for those of us who, like me, can't afford the latest 70 MHz 80586 box. Although Microsoft provides a dialog editor as part of the SDK, this program isn't capable of modifying resources in place within a .EXE file. The Toolkit, however, allows you to do just that. You can run your program, modify a dialog box, run the program again, and see the changes immediately. But don't get too ambitious - you must not try changing the resources of a program while it's actually running. If you do then, in the words of the manual, 'the system will throw up its hands and walk out on you.'

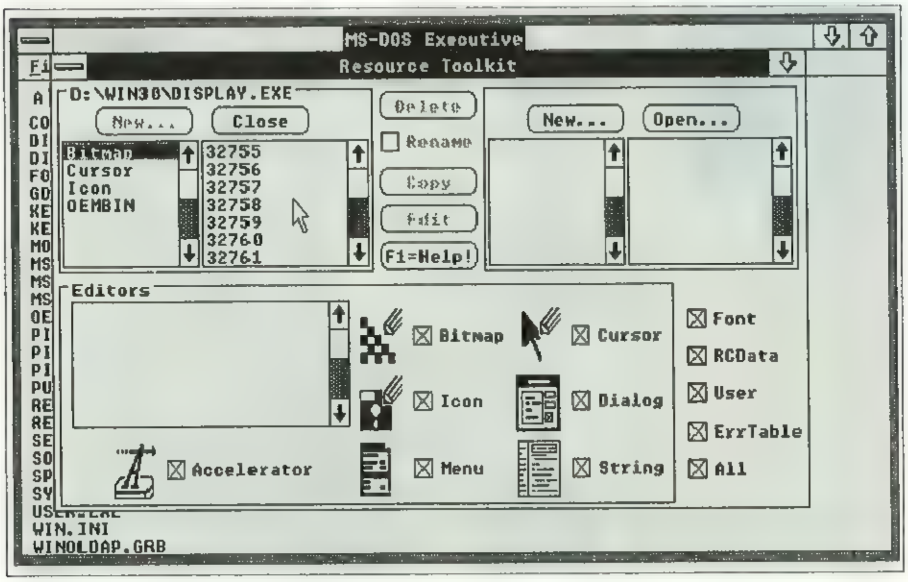


Figure 1 - Main screen of Resource Toolkit

Using the Toolkit

The Resource Toolkit comes on two 5.25 inch disks together with a slim manual. Installation consisted simply of inserting the first disk, typing 'INSTALL' and following screen instructions. You end up with two main files, WRT.EXE and WRT.IMG, in a directory of your choosing. The WRT.IMG file is no lightweight, weighing in at over 400 KB. There is also a special font file, called RESYM.FON, which contains the special characters used by the Resource Toolkit program.

I found it quite difficult to get to grips with the manual, mainly because of the laboured way in which it explains the function of every possible control of every possible dialog box. After all, the Toolkit is aimed at professional programmers, not computer novices. Having said that, I've no doubt that all the necessary information is there, if you can track it down.

To run the Toolkit program, you just double-click the WRT.EXE file in the normal way. After an initial welcome screen, you'll see the main dialog shown in Figure 1. The top half of the dialog box contains two 'picker' lists, each of which enables selection of a specified resource file and, subsequently, a resource within that file. Normally, you would choose one resource file, then proceed to edit its existing resources or add new ones. But you can usé the two pickers to select two different files and exchange ('move') resources between them. For this reason, Whitewater has named this main dialog 'the Mover screen' (a slightly desperate piece of nomenclature, presumably devised after a certain amount of head-scratching).

The bottom half of the main dialog allows you to specify the resource types which are currently of interest. Any 'unclicked' re-

‘ WELL DRAWN! ’

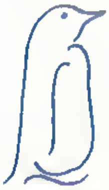
.EXE Magazine, March 1990

You draw the picture on the screen, AutoGraphic does the rest!

AutoGraphic is a code generating tool. Using the mouse, you design your graphics display. When you've finished, AutoGraphic creates the code which describes that image. AutoGraphic comes in two versions, one of which writes Borland Turbo C V.2.0, and the other, Turbo Pascal V.4.0 and above.

‘AutoGraphic impressed us. It represents a real productivity boost for Turbo C programmers.’ *Paul Roddick, Technical Manager, Borland International Ltd.*

- AutoGraphic simplifies the production of graphic code and saves hours of development time.
- All usual drawing devices are available. Circles, rectangles and lines can be rubber-banded in any IBM colour. Enclosed areas can be filled with solid colour or patterns.
- AutoGraphic has a multiple undo facility which allows you to backtrack through the last twenty five steps.
- Creates resolution-independent code, supports Hercules CGA, EGA and VGA screens.



ONLY £49.95

Orders to

PenguinSoft Ltd.

Easterton by Monikie, Broughty Ferry,
Dundee DD5 3QL
or telephone

02416 391

BARCLAYCARD VISA Access

A Microsoft compatible mouse and driver is required to run AutoGraphic

CIRCLE NO. 922

DON'T GIVE IT AWAY!

Software Developers! Protect Your Software Against Unlicensed Use.

From **Link Computer** Denmark, a remarkable piece of pure software that will lock a DOS .com or .exe file to a diskette, machine or network so that you may copy (or back-up) to your heart's delight, but you can still only run the program on the designated medium.

It's called **Cop's Copylock II** and it's clever stuff. You get an unlimited number of protections per purchase and its also uncrackable. Forget your favourite copying programs - they won't work on this. Forget Option Board and Bit Copiers - we dealt with them years ago.

If you don't want to dabble with dongles and aren't so keen on your software being freeware, you ought to take a look at this.

Ring Dino Cano on
Tel: 0763 263 073 Fax: 0763 262 620

Link Computer ApS.

CIRCLE NO. 923

quikcad

An unique opportunity for software developers to utilise the source code of a high quality two dimensional CAD package.

Commenting on the original version of quickcad, PC Magazine said *"As a comprehensive set of tools for technical or architectural drawing (quickcad) has a great deal to offer. The ability to alternate between precise keyboard input for exact angle and line drawing, and cursor control for speed drawing, together with the auto-dimensioning capabilities of this package, make it an attractive proposition ... At £895 it is one of the cheapest systems on the market ... that competes directly with mini and mainframe CAD systems".*

Not only have EMTEC relaunched this outstanding package in its entirety for just £69 (inclusive of VAT), you can also purchase the source code for an additional £621 (inclusive of VAT).

SPECIAL OFFER
Software and Source Code
ONLY £690

quickcad provides:

- **ease of use:** with a simple command language
- **drafting power:** with an extensive range of graphical construction and editing facilities
- **flexibility:** for a variety of applications including architectural plans, mechanical drawings, curcuit diagrams, flow charts and technical illustrations
- **speed:** for fast and efficient drawing production
- **layering:** each drawing can consist up to 200 layers for complex and highly accurate designs

IBM PC compatible, quickcad supports a wide range of graphics printers, plotters, digitizers and mice.

I wish to order quikcad software □ and source code □
I enclose my cheque including £1.50 carriage for £________ or
charge my Access □ Visa □ No: Expiry date ________

Name ____________________

Address ____________________

____

Tel: ____________________

EMTEC

ENGINEERING & MAPPING TECHNOLOGY LIMITED
St John's Innovation Centre, Cowley Road, Cambridge CB4 4WJ
Tel: 0223 421175 Fax: 0223 420844

.EXE 5/90

CIRCLE NO. 907

source types will be filtered out of the various picker lists which appear. The scrollable list box towards the bottom left allows you to bring different editor windows to the front, for those cases where you re editing more than one resource type at once.

The Toolkit supports the seven most common Windows resource types: bitmaps, icons, menus, cursors, dialog boxes, strings and accelerators. There is a built-in editor for each type, which understands the resource format and provides appropriate editing facilities. For example, Figure 2 shows the dialog editor in use.

If you have a resource which doesn't fit into one of these seven categories, the Toolkit will display it in the 'picker' list associated with each resource file, but you won't be able to modify it. This is likely to be a problem for Windows programmers who have made extensive use of the RCData facility provided by RC.EXE, the Microsoft Resource Editor. (If you want to define a new application-specific resource type, RC allows you to do this by giving the resource a type of RCData. It's a sort of generic data type, and can be used for any purpose.) I feel that the ability to edit unrecognised resource types is a significant, and unnecessary, omission from the Resource Toolkit. It would have been better to provide a default editor for unknown resources, which could let you patch your files, in hexadecimal, on a byte by byte basis. Even a clumsy tool is better than nothing.

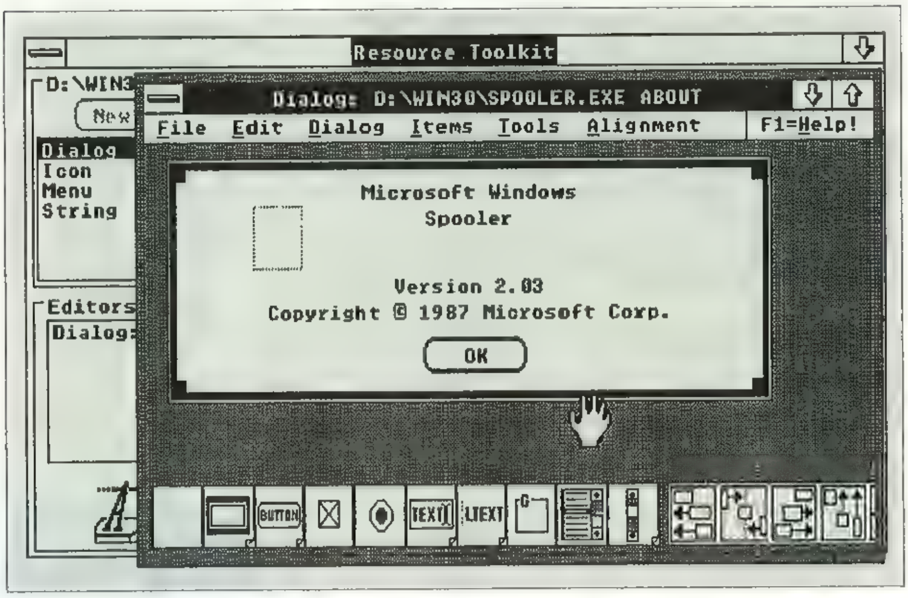
Direct support for header files is a nice feature of the Toolkit. This allows you to associate a header file, containing a list of `#defines`, with a resource type. Once this is done, you can refer to resources by name rather than by numeric ID.

The Toolkit supports resource files with the following extensions:

- .BMP (Bitmap resource data.)
- .CUR (Cursor resource data.)
- .ICO (Icon resource data.)
- .RES (Resource data compiled with the Resource Compiler, but not yet appended to the end of the Windows application file.)
- .EXE (Resources contained within the final, executable application.)

You can also use the Toolkit to generate .RC files (these are the input scripts which are used as 'source code' by the resource compiler).

Figure 2 - The dialog editor in use



Microsoft Windows
Spooler

Version 2.03
Copyright © 1987 Microsoft Corp.

OK

Because the Toolkit allows you to edit compiled programs, you can look at the resources to be found inside existing Windows applications. The Microsoft Windows display driver contains a set of bitmaps that correspond to the various little graphical 'gizzmos' of the Windows interface: the minimise and maximise buttons, the arrows used for making scroll-bars, and so on. Microsoft puts all this stuff into each display driver so as to provide a set of gizzmos (sorry, I can't think of a better word) which look good for the particular screen resolution used by that driver. I thought it would be interesting to make my copy of Windows look a little more like the Motif interface. I renamed my copy of DISPLAY.DRV to DISPLAY.EXE (so that the Resource Toolkit could see it) and used the Whitewater Toolkit to edit the various resources inside the display driver. I renamed the driver back to what it was, re-installed Windows, and hey presto! instant Motif... Well, almost. Figure 3 shows one of the display driver gizzmos in the bitmap editor, ready to receive the Motif treatment.

Editors

The editors provided in the Resource Toolkit are very full-featured. They're a bit like miniature painting/drawing applications in their own right. The bitmap editor, for example, provides tools to change the pen width, draw free and constrained lines, filled and unfilled rectangles, polygons, ellipses, and so forth. The dialog editor is especially comprehensive. There is a set of alignment functions, which allow you to do things like selecting a group of dialog controls and aligning them along one edge, or making them all the same size, or even spreading them out evenly along the horizontal or vertical axis. Funnily enough, it's this one feature of the dialog editor which, more than anything else, would make me want to buy the Toolkit. I get really irritated if I see that one of my dialog buttons is a pixel higher than its neighbour, and used to waste a lot of time, peering at the screen, getting things lined up correctly.

Performance

While the Resource Toolkit contains a number of features which are worth having, I was a little disappointed with the performance. On a 640 KB machine, without EMS memory, the system ran rather slowly, even though I was using a 2 MB RAM cache. The Resource Toolkit takes a lot of memory and, in fact, Whitewater recommends a minimum of 1 MB of expanded memory. While I haven't had much experience of using Actor, I suspect that the performance problem was partly due to the fact that the package was written using version 1.2 of that language. Using EXEHDR, a utility that comes with the Windows SDK, I discovered that the WRT.EXE program contains only one code and one data segment. Both of these segments are fixed and are the best part of 64 KB in size. This is just not acceptable practice in Windows programming. It's tantamount to declaring war on the Windows memory manager. Presumably version 2.0 of Actor, which is claimed to provide more usable memory, will go some way towards reducing this overhead.

Conclusions

While using the Whitewater Resource Toolkit, I found myself continually comparing it with a program called ResEdit, which runs on the Apple Macintosh. For those unfamiliar with the Mac, ResEdit performs a very similar job to that addressed by the White-

PC Magazine Award winner for TECHNICAL EXCELLENCE Development Tools category

INTRODUCING THE REMARKABLE MULTISCOPE DEBUGGER FROM LOGITECH

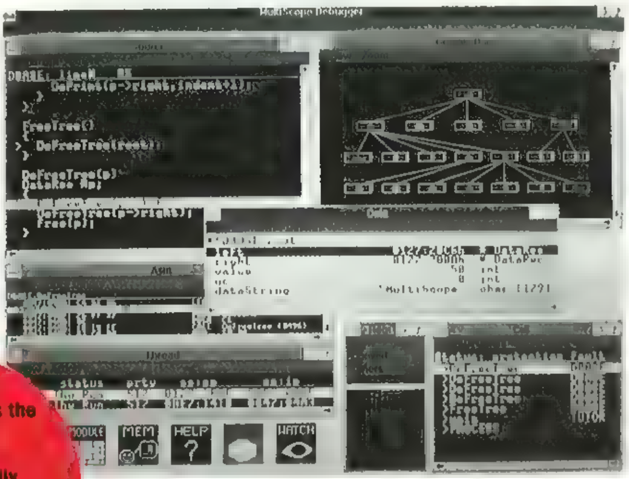
More Debugging Power For OS/2

Logitech's new MultiScope ™ Debugger is quite simply the most powerful and advanced debugging tool available.

This state-of-the-art program, supplied and supported by QA, represents a milestone in debugging technology. Not only will the MultiScope ™ Debugger help relieve the burden of tedious debugging, it is also the only debugger designed to take full advantage of the extra functionality of OS/2.

There's even run-time and post-mortem debugging options, allowing examination of your program both during execution and after a run-time exception.

Never before will you have experienced such a clear insight into your program's inner workings.



```
DoPrint(p->right,indent+1);
  }
}

FreeTree()
{
> DoFreeTree(root);
}

DoFreeTree(p)
DataRec *p;
{
    DoFreeTree(p->right);
    Free(p);
>
```

```
right                0127:0000   * DataRec
value                      50     int
                            0     int
dataString        'MultiScope     char [127]
```

```
status   prty
```

Logitech's MultiScope ™ is the ONLY debugger to fully utilise the OS/2 programming environment

USER INTERFACE

MultiScope uses either a Presentation Manager or an OS/2 text mode user interface to provide you with the perfect interface for your requirements. Both execution modes are compatible with the Common User Access of the IBM Systems Application Architecture for maximum ease of learning and use.

DEBUG MULTIPLE LANGUAGES

- IBM C/2
- IBM Pascal/2
- IBM Macro Assembler/2
- Logitech Modula OS/2
- Microsoft C
- Microsoft Pascal
- Microsoft Macro Assembler

2 DEBUGGING OPTIONS

- **Run-time**

MultiScope allows you to monitor the execution of a program with conditional or unconditional breakpoints, and memory or symbolic watchdogs. While you are stepping in your program execution, MultiScope allows you to modify memory or data contents, suspend or resume thread execution.

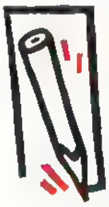
- **Post-Mortem**

This debugging mode gives all the MultiScope debugging power for examining a program after the occurrence of an OS/2 protection trap, stack overflow, or other run time exception.

GRAPHICAL DATA REPRESENTATION

Only the Logitech MultiScope ™ Debugger permits the graphical representation of your program's data structures. This feature, which operates under Presentation Manager, allows you to explore your dynamically allocated data structures and their internal relationships.

MAXIMUM PERFORMANCE IN TWO MODES



- **Presentation Manager Mode**

MultiScope takes full advantage of Presentation Manager's graphics, multiple fonts, overlapping windows and standard user interface.

- **OS/2 Text Mode**

MultiScope offers a text mode user interface with a very easy-to-use windowing system for the EGA and VGA graphic adaptors. This execution mode is the perfect solution for debugging Presentation Manager applications.

Pull down menus that respond to single-key commands make your debugging faster and easier. Create your own custom configuration of these windows to see exactly what you need. Reduce currently unused windows to icons for easy recall.

MULTIPLE VIEWS

MultiScope gives you 13 different views into your program:–

Source View Look at source code, set breakpoints, find symbols.

Data View Display and modify local and global symbolic data, records, arrays, and set watchpoints.

Graphic View In PM mode, see a graphic representation of any selected data structure.

Assembly View See the assembly decode of the program, complete with the display of corresponding source lines and local and global data.

Thread View Observe threads belonging to your application, and suspend threads, resume threads and update views according to new threads.

Breakpoint View Display the current breakpoint locations, count and limit.

Watch View Display the current watch variables or memory location.

Register View Display and modify the current values of the CPU registers.

Output View Examine the application screen.

Module View Choose between different data object modules of the application and view the corresponding source or data.

Call View Display the source position, local data and parameters of a particular call.

Memory View Display raw memory in various formats; byte, char, text, unsigned word, signed word, address, double word, real, etc.; modify memory contents; set watchpoints.

Log View Log debugger events.

SEND YOUR ORDER TODAY!

ONLY £239

+ VAT including UK delivery

MORE FEATURES

- **Execution Modes**

MultiScope allows you to execute your program until:–

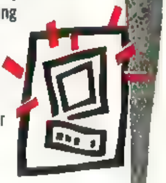
- the next conditional breakpoint is true
- the next high level instruction
- the next line
- the next assembly instruction
- the return from the current procedure
- the next call from a current procedure
- the next dynamic link load
- the next child process load
- an OS/2 protection violation or other runtime exception
- the next watchdog activation

- **Evaluate Expressions**

MultiScope allows you to evaluate C, Pascal and Modula-2 expressions using symbols defined in your program.

- **Function Calls**

MultiScope is the only OS/2 debugger which can execute calls to your own code during the debugging session. Use your own procedures to print out data structures, or experiment with calls with different parameter values.



- **Child Process Debugging**

When your program executes another program, MultiScope allows you to continue debugging the child process.

- **Dynamic Link Libraries**

The full power of MultiScope is available for debugging your dynamic link libraries.

- **Remote Debugging**

MultiScope allows you to debug a program executing on another computer through a serial line. This feature allows your application to keep all the resources of its computer for its execution.

- **Dual Monitor Debugging**

MultiScope allows you to use a dual mode monochrome monitor for the display of the debugger views while your program is executing on your primary colour monitor.

SYSTEM REQUIREMENTS

- IBM XT/286, AT or 100% compatible
- IBM PS/2 model 50, 60, 70, 80 or 100% compatible
- OS/2 Standard Edition 1.1 or Extended Edition 1.1
- One high density floppy drive and a hard disk

TRY MULTISCOPE FOR YOURSELF

Whilst we've highlighted the main features of this powerful debugging tool, quite honestly the only way to find out about MultiScope is to try it. Call us at QA for a free demo disk containing a complete functioning debugger with tutorial program.

HOW TO ORDER

Call Jan Bevans, Products Group, QA Training Ltd, Cecily Hill Castle, Cirencester, Gloucester GL7 2EF on (0285) 655888 or fax (0285) 650537. Company orders, cheques and Access/Visa welcome. MultiScope costs £274.85 (£239 + VAT) including UK delivery. Call for volume discounts.

QA

QA Training Ltd, Cecily Hill Castle,
CIRENCESTER, Gloucester GL7 2EF
Tel: (0285) 655888 Fax: (0285) 650537

CIRCLE NO. 908

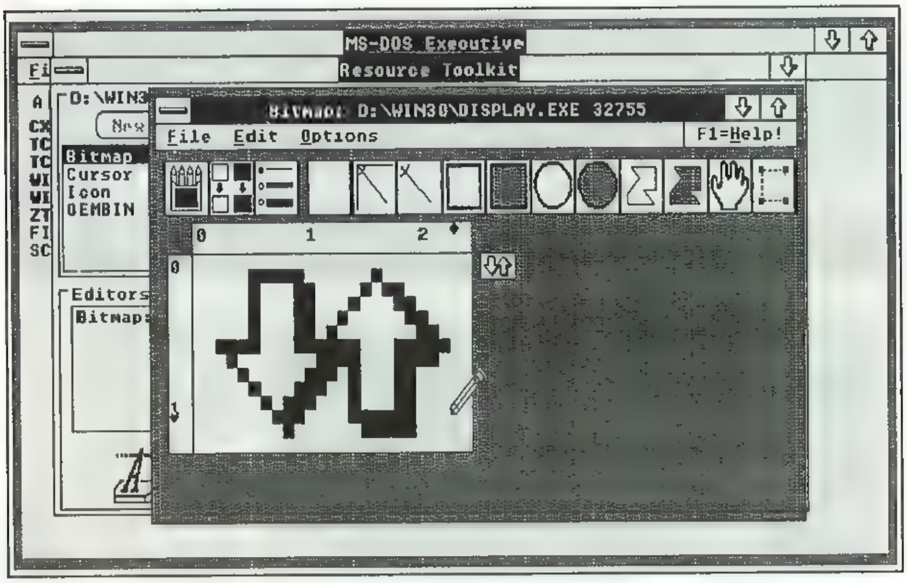


Figure 3 - Editing the Windows display driver

water product but, undeniably, it boasts a much more intuitive interface. ResEdit presents resources in a cleaner, more obviously hierarchical fashion: double-clicking on a filename displays a new window, showing a list of resource types within that file. Double-clicking on one of these resource types yields yet another window containing a list of all resources of that type. Finally, double clicking on an individual resource invokes the resource editor appropriate to that resource type. With ResEdit you can instantly move between these different 'levels'. If you need to move resources from one file to another, ResEdit allows you to simply select the required resources, perform a Cut or Copy operation, and then Paste the resources into the destination file of your choice. This does away with the need for two separate 'picker' displays. ResEdit also scores in the area of extensibility. It is possible, on the Macintosh, to write your own resource editor modules and add them to the ResEdit program file. This sort of thing could be accomplished under Windows by creating a .DLL library of custom resource editors.

However, despite the unexciting documentation and the shortcomings of the user interface, there's no doubt that this package is a vast improvement on the dialog editor provided by Microsoft. It should drastically reduce the amount of time required to develop resources for Windows applications and should find a place on the book shelf of every Windows programmer. My only real reservation about the usability of the program concerns the large amount of memory it consumes. Windows can easily slow to a crawl while you're using the Resource Toolkit. Any serious user will need to invest in some EMS memory. EXE

Dave Jewell works for the Sign Express Group, Basingstoke, as a Senior Consultant. He can be contacted on CIX as 'djewell'. The Whitewater Resource Toolkit costs £145 and is available from Neow Ltd (0628 668334).

PCL 4.0 instead of BASIC, PASCAL or C for DOS and OS/2

Produce blazingly fast applications in record time. PCL has 4GL-like super functions but combines them with the execution speed of the best compilers.

PCL programs are easier to write than BASIC, more readable than C and as well-structured as PASCAL.

- **Supports 4 interrupt driven comms ports, 2-19200 baud.**
- **Optional SQL interface to ORACLE.**
- **Dynamic record structures.**
- **Built-in windowing & menu system.**
- **Multi-tasking & shared variables in OS/2.**
- **Ram-resident in DOS.**
- **Background tasking in keyboard wait-state in DOS and OS/2.**
- **Array, time/date & extended scientific maths, 80x87 support.**
- **Fastest sort, search, text & file handling in the business.**
- **Run-time execution of source lines.**
- **Network support.**
- **Licence free run-time module.**

DOS	**Demo Disk**	£5
DOS	PCL 4.0	£195
DOS	Oracle/SQL tool kit	£65
OS/2	PCL2 2.0	£220

Postage included. Add 15% VAT. Specify diskette type: 3.5" or 5.25" and operating system (DOS or OS/2).

Orders for the demo diskette must be prepaid. All others are dispatched with invoice, for payment within 30 days.

CALEND

PO Box 94, Twickenham, TW2 6DD, England
Telephone: 081 894 7409 FAX: 081 755 0670

Borland's Paradox Engine

The Paradox Engine is a library of routines, callable from C or Turbo Pascal, that lets you read and write Paradox files. Andy Redfern tried it out.

When Borland first introduced Paradox, the critics chimed in unison 'No chance'. No one really believed that Borland could break the Ashton-Tate stranglehold on the database sector, without making it at least dBASE compatible. But both incompatible and successful it has been.

Paradox is a relational database management system (RDBMS) that allows tables (Paradox's name for database files) to be connected together explicitly through links. To provide full control, and to shield the user from the complexities of Paradox, PAL (Paradox Application Language) is included. This provides the programmer with a simple high-level language interface that takes full advantage of all Paradox's database and networking capabilities. Like all application building languages, though, it has distinct limitations.

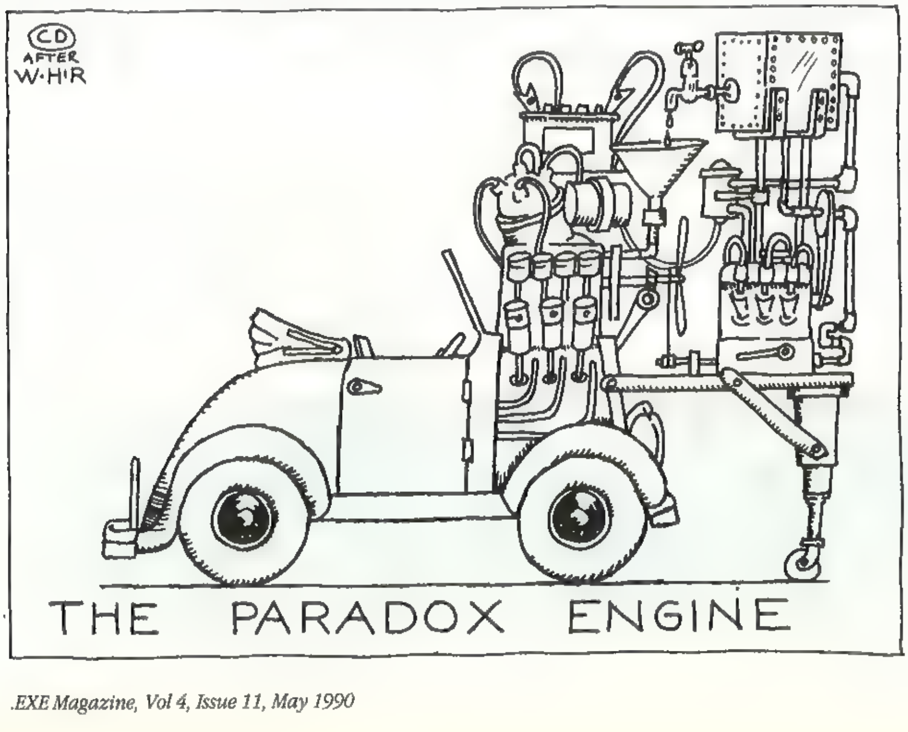
First, PAL is an interpreted language, so some complex operations can take a considerable length of time. Second, there are some things that PAL just can't do. For example, if you were writing an electronic point-of-sale system that worked with Paradox, it would be impossible to link the bar code reader directly with the database. An intermediate file would have to be created and processed at a later date.

To provide programmers with a solution to these problems, Borland have launched a C language interface to Paradox called the Paradox Engine.

The Engine

The Paradox Engine is basically an applications programming interface (API) that allows, through a series of 70 function calls, direct access to Paradox tables and indexes.

The API allows you to create, read and write Paradox tables, records and fields. It provides support for password protection, encrypted tables, date encoding, searching and error handling. It can create and manipulate tables through serial communications, by downloading from mainframes, or from other external devices that cannot normally be reached by PAL.



It also provides network support with file, table and record locking. This means that both Paradox and Paradox Engine applications can gain access to a single table across a network without the integrity of the file being lost. The Engine, like Paradox itself, supports IBM Token Ring, Novell, 3COM, Banyan, Torus, AT&T Starlan and any other network that is 100% compatible with DOS version 3. The Engine applications can be run as stand-alone programs, or can be called directly from PAL using the RUN or, in most cases, the RUN BIG command. The API can be called from Turbo C, Microsoft C and, very soon, Turbo Pascal.

The Engine already forms the file control mechanism in two other Borland products - SideKick for Presentation Manager, and Quattro Pro. This allows Paradox, SideKick and Quattro to share files on a single server across a network.

Borland clearly sees this as an important element in its corporate strategy and are trying make the Paradox file format a standard for network applications. The launch of the Engine so that other programmers can use it, makes an obvious contribution to this strategy. Nantucket have already announced that they will be including support for Paradox tables in Clipper 5.

Installation

Like virtually all Borland's products, installation is simply a matter of typing `A:INSTALL`. After specifying the drive and directory where you want the Engine to reside, the library files and example applications are copied across.

Applications that use the Engine library can, at present, be written in Turbo C version 2.0 or Microsoft C version 5.1. For the purpose of this review, I tested that the Microsoft C compiler coped with the sample applications, but all serious testing was done using Turbo C.

D DESKEY

The two sample applications that are included with Engine provide an excellent reference for exactly how to write your own application - a detailed explanation of how they work forms the major part of the manual. My only criticism is that a Turbo C .PRJ or MAKE file was not included. This would have greatly simplified the task of getting the sample applications up and running.

The first example application is an ASCII Text to Paradox Table file conversion utility, while the second is a network phone directory called FONEDEX. The latter is actually quite a useful little application which, with the addition of a better front end, could make the basis of a neat network product. This program excellently shows how to pre- and post-process the raw data to make it suitable for Paradox tables to handle, and provides access to the basic error handling routines.

As well as the sample applications, a whole host of example files are included. These are simple programs that show, in a few lines of code, how to use one aspect of the Engine. The examples are also listed in the manual next to the relevant function reference. This not only allows for quick reference, but also provides a mechanism for including working sections of code in your own programs.

Compiling an application with either Turbo or Microsoft C is performed in the normal way for each compiler, except that the large memory model must be specified. If you are using the Microsoft compiler, the default stack size must be increased. Currently, the Engine adds between 60 and 100 KB to the size of an application.

The Engine in Use

Creating an Engine application is a fairly simple, if a little long winded, process. The function prototypes and data structures for the Engine are all in a single header file called PKENGINE.H which provides the four major global handles that the Engine uses - record, table, field and lock. It also includes `#defines` for the different modes and arguments required by various functions, thus making the final code a little more readable.

The first call is normally to the Engine initialisation routine `PXInit`, unless you are creating a network application when `PXNetInit` should be used instead. The only call that you can make before initialising the system is to get or set the default information about how the Engine should work.

In most cases, the defaults will be satisfactory, but there are still pitfalls to avoid. For example, if you use a copy of Paradox to create a table and an index, you will only be able to read the index providing the default settings for sort order are the same in your application as they were in Paradox.

Once the Engine has been initialised, you can begin the serious work of manipulating Paradox tables and indexes. To open a table, `PXTblOpen` is called. This requires a pointer to a table name, and a table handle. You can also indicate whether to use an index, if there is one, and whether or not changes should be saved as they are made or whether they should be stored until the table is closed. Then finally, a record buffer needs to be opened for the table. This is where the Engine keeps the current record and where the results of any search will be found.

A typical access to a Paradox table is shown in Figure 1. The program searches the database for record number 64 and then retrieves the data and prints it out. The

```
#include <stdio.h>
#include "pxengine.h"

RECORDHANDLE rechandle;  /* record buffer */
TABLEHANDLE tblhandle;  /* table handle */
FIELDHANDLE fldhandle;  /* field handle */

const char *fname[2] = { { "HROYALP.DB" },
   { "HDEMYP.DB"  } };

const char *fieldno[5]= { { "Record-Number" },
   { "Field-2" },
  { "Field-3" },
  { "Field-4" },
  { "Field-5" } };

int main(void);     /* Prototypes  */
int Error(int);

int main(void)
{
 int fileno=0;
 int record=64;
 int i;
 double tmp;

 if (Error(PXInit())) exit(1); /* Initialise engine */
 clrscr();
 Error(PXTblOpen((char *) fname[fileno],&tblhandle,0,0));
 rror(PXRecBufOpen(tblhandle,&rechandle));
 Error(PXFldHandle(tblhandle,"Record-Number",&fldhandle));
 Error(PXPutShort(rechandle,fldhandle,record));
 Error(PXSrchFld(tblhandle,rechandle,fldhandle,SEARCHFIRST));
 Error(PXRecGet(tblhandle,rechandle));
 for(i=1;i;i++) {
  Error(PXFldHandle(tblhandle,(char *) fieldno[i],&fldhandle));
  Error(PXGetDoub(rechandle,fldhandle,&tmp));
  printf("Field %d  %4.0f\n",i+1,tmp);
 }
  return(Error(PXExit()));
};

/* Simple error handler */
/* Prints out Engine error message and returns */
int Error(int rc)
{
  if (rc != PXSUCCESS) /* Defined in header file */
   printf("ESTIMATE: Paradox Engine: %s\n",PXErrMsg(rc));
  return rc;
};
```

Figure 1 - Accessing a Paradox Table

Initialisation and shut down
Functions to initialise and shut down system on both networked and stand alone machines. Also allows Paradox default information files to be loaded and saved.

Table manipulation
Allows Paradox tables to be opened, closed, added to, copied, deleted, emptied, named and renamed. Also allows the table size, and whether or not the table exists, to be checked.

Record manipulation
Allows records to be appended, deleted, loaded, inserted and updated. It also allows the new current record to be set to the first, last, next or previous record. Provides control of the memory resident record buffer.

Field Manipulation
Allows the field type, handle and name to be set, and provides a mechanism for loading and saving each of the different field types.

Index control
Provides a simple mechanism for adding and deleting indexes, and reindexing a file.

Date control
Encodes or decodes a date from Paradox format to calender format (day, month, year).

Search operations
Allows a search of the index or the Paradox table itself.

Password and security facilities
Provides password protection and encryption to tables.

Informational operations
Gives access to the network environment information and the status of field, record and table handles.

Network and Concurrency support
Provides network support with file, table and record locking.

Error handling Operations
Full reporting of all errors through detailed messages.

Figure 2 - Main types of function provided by the Engine

program works as follows. First, a field handle is assigned to a named field - `Record-Number`. A short integer is then put into the record buffer and a search is carried out. The record is then retrieved using `PXRecGet`. Then, by reassigning the field handle to the different fields in the record in turn, the data contained in the specific record, in this case four floating point numbers, can be retrieved from the record buffer.

One problem with this example is that the type and structure of the Paradox table must be known before the program runs. If the first field isn't a short, or isn't called `Record-Number`, the record will not be found. To enable unknown tables to be examined, the Engine provides a set of informational functions.

`PXRecNFlds` returns the number of fields in each record of the table, while `PXKeyNFlds` shows which of them is a key or indexed field. Using `PKFldType`, the field type can be ascertained. Now the correct type of `PXRecPut` and `PXRecGet` can be called. As you can see, everything is possible, eventually.

Like the functions that come with any C compiler, the way to see the Engine is as a set of building blocks from which you can make your own less verbose facilities. For example, I wrote a function called `TblSearch` that was called with a table handle, a record handle and a search structure. It then did all the necessary `puts` and `gets`, dealt with any error conditions and returned the answer in another structure. The functionality was the same, but the verbosity of the finished program was reduced completely.

Speed

It's not easy to judge the speed at which an Engine application runs. As the Engine is initialised, it opens a buffer where it stores all its reads and writes. Unless specifically stated, the Engine will post-write most changes, and keep copies in memory of all the most recently read records. In a sample application I wrote, initialising the Engine, opening eight different files and searching for a number of items took less than a second. Writing the results of the test (approximately one tenth of the data) back to disk using a standard `fprintf`, took almost exactly the same time. This is a very fast file handler.

Networking

Networking is so simple that it almost seems worth writing every application as though it were going to run on a network. The only penalty is that the network handling routines add another 30 KB to the .EXE file size.

Basically, the only changes that need to be added are in the initialisation procedure and in any write/amend operations that are invoked. `PXNetRecLock` and `PXNetRecUnLock` are used to lock and unlock records when they are being modified across the network. If a file is only infrequently used by other users, table locking rather than record locking can be employed, because this is faster. Once that has been done, the application will run safely on any of the networks supported. I was impressed at how simple this was to achieve, and believe it will encourage developers who have shied away from networks to take another look.

Documentation

The documentation consists of a single 250 page manual which covers installation, examples and reference. The reference and example sections are excellent, but I felt the introductory section could have explained more clearly how the Engine worked, and the reasons for the way in which it does things. A few words of the methodology used would have helped to alleviate the 'There *must* be a simpler way' feelings that were aroused in me every time I saw the list of functions I needed to call, even to achieve the simplest of things.

Conclusion

My only real criticism of this package is the verbose list of function calls that are needed to do anything more complex than opening a table. When I put this to Phillipe Kahn, he commented that a 'higher level of interface' was in the pipeline. This would be a set of function calls that would shield the programmer, if he or she wanted to be shielded, from the mechanics of buffers and handles. A more typical 'search, and return with the answer' style of interface would be provided. Details of this new version are expected to be released at the Paradox Developers' Conference in Monterrey later this month. This also may account for the delay in launching the Pascal version of the product.

In general, this is an excellent product. It is a robust package that provides Paradox application developers with a powerful way of overcoming the limitations of PAL. It should allow application developers to develop more intelligent systems that can integrate with real world and real time events like scanners, bar code readers, modems and other applications on networks and stand alone PCs. But the advantages don't stop there.

Imagine you were developing an application to run on a network. Wouldn't it be simpler if all the network file handling and contentions were handled for you? I used the Engine to write an application for Collins Publishing. It was a stand-alone application that accessed upwards of 20 tables. It then performed various calculations and manipulations of that data. If, tomorrow, Collins decide to run the application on a network instead of a stand-alone system, half an hour's programming would convert the application.

Not only is the Engine an excellent product for Paradox application developers, but it could also become an off-the-shelf solution to creating network specific applications. Borland have always wanted their file formats to become de-facto standards within the industry; selling a powerful networking engine that allows access by most levels of programmer could, perhaps, be the way of doing it. .EXE

Andy Redfern worked as a writer for Personal Computer World Magazine, and is now a freelance journalist specialising in PCs and programming.

PRODUCT DETAILS:

Equipment required:
IBM PC, PS/2 or compatible
512 KB memory (640 KB recommended)
Hard disk
DOS 3.0 or higher
Turbo C V2.0 or Microsoft C V5.1
Paradox V2 and V3 tables supported

Database Limitations:
Maximum 64 open tables
256 KB maximum swap buffer
128 KB maximum record buffer
128 KB maximum file locks
2 billion records per table
255 fields per record
4000 bytes per record (only first 1350 indexed)

Licenses:
Applications written using the Paradox Engine API have an unlimited, royalty-free run-time licence.

Manufacturer:
Borland (0734 320100)
Price: *£325*

Menu-IX | System-IX | DBMS-IX | Sysix | AI | Spell | Merge | Lock | GURU | Crypt | Tawk | Tgrep

Cut | More | Term Ctl | Where | Sort | Grep | Idle | CB Trees | Evaluate | Awk | Banner | C Source | Fgrep | Help | Head | Tail

Xenix | Xenix Net | Unix | Interactive | System V | TCP/IP | DOS | Token-Ring | OS/2 | Ethernet | Novell Lan | Mic Net | Aix | VP-IX | VMS | QMS

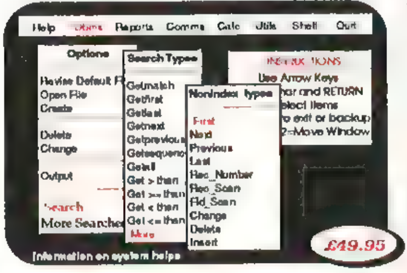
Leading Product Prices Slashed

XENIX UNIX & DOS Users

Release the POWER of your UNIX XENIX DOS Operating System

MENU - IX

XENIX UNIX DOS User friendly interface



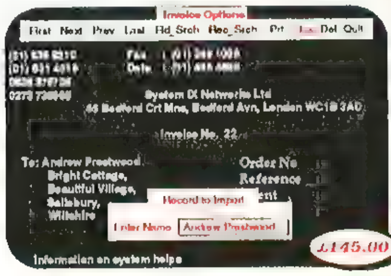
```
Help  Options  Reports  Comms  Calc  Utils  Shell  Quit
Options: Revise Default F  Open File  Create  Delete  Change  Output  Search  More Searches
Search Types: Getmatch  Getfirst  Getlast  Getnext  Getprevious  Getsequence  Getall  Get > than  Get >= than  Get < than  Get <= than  More
NonIndex types: First  Next  Previous  Last  Rec_Number  Rec_Scan  Fld_Scan  Change  Delete  Insert
INSTRUCTIONS: Use Arrow Keys ... char and RETURN ... select items ... to exit or backup ... 2=Move Window
Information on system helps
```

£49.95

Powerful, friendly, colourful, complete menu system that *you* can customise to exacting requirements.

SYS - IX

XENIX - UNIX - DOS - 4GL Database



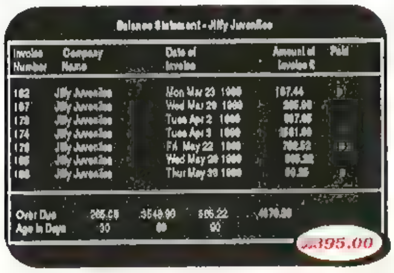
```
Invoice Options
First  Next  Prev  Last  Fld_Srch  Rec_Srch  Prt  ...  Del  Quit
System IX Networks Ltd
46 Bedford Crt Mns, Bedford Avn, London WC1B 3AD
Invoice No. 22
To: Andrew Prestwood   Order No
Bright Cottage,        Reference
Beautiful Village,
Salisbury,             Record to Import
Wiltshire              Enter Name  Andrew Prestwood
Information on system helps
```

£145.00

A powerful & complete applications generator for forms & database management, includes Menuix.

SYSTEM - IX

XENIX - UNIX - DOS - 4GL Reporting



```
Balance Statement - Jilly Juveniles
Invoice  Company         Date of          Amount of  Paid
Number   Name            Invoice          Invoice £
162      Jilly Juveniles Mon Mar 23 1990  167.44
167      Jilly Juveniles Wed Mar 28 1990
...
Over Due   265.05  3549.99  565.22  4670.00
Age in Days  30      60       90
```

£395.00

Powerful toolset for reporting database file systems under Menuix & Sysix

Special Offer on Sys-IX

Free Application Modules

Bank Reconciliation
Integrated Customer Name & Address
Invoice & Ledger Database

Free Report Modules

Monthly turnover
Monthly VAT, monthly gross profit
Auto statement production with
Outstanding aged analysis

Free Utilities

General User Reporting Utility
Desktop Calculator & Calendar
Artificial Intelligent Help System
Programs is designed to enable
you to use your own stationary.

Prices exclude VAT & Delivery

SAGESOFT	
Sterling Accountant	£169
Sterling Accountant Plus	£279
Sterling Bookeeper	£75
Ster. Financial Controller	£349
Ster. Job Costing	£150
Ster. Network Accountant	£270
Ster. Net. Accountant Plus	£365
Ster. Net. Fin. Controller	£495
Ster. Net. Job Costing	£185
Ster. Network Payroll II	£185
Ster. Payroll II	£145
CAD & DESIGN	
Intro Training Course	£75
Drafix 3D Modeler	£247
Drafix Ultra	£475
Report Writer	£135
Symbol Lib Arch/Elec/Mech	£125
Turbo CAD v1.7	£109
COMMUNICATIONS	
Procom Plus	£47
Crosstalk XVI v3.62	£99
PC Anywhere III v3.10	£99
Carbon Copy Plus v5.1	£119
Smartmove 470	£157
Chit Chat Dual	£79
DATABASE SOFTWARE	
dBase III Plus v1.1	£325
dBase IV	£369
dbase IV Devel Edition	£599
Sys-IX Xenix/Dos/Unix	£195
System-IX Xenix/Dos/Unix	£395
R-Base Dos v2.11	£377
R-Base Tools	£79
Clipper Summer 87	£325
Documentor	£193
Nantucket Tools	£155
Retrieve III Single User	£119
Professional File v2.0	£219
DataPerfect Sngl N/W v2	£347
DBXL v1.3	£240
QuickSilver v1.3	£390
MICROSOFT	
Basic Compiler v6	£169
C Compiler v5.1	£199
COBOL Compiler v3.0	£388
Fortran Compiler v5	£219
Macro Assem v5.1	£95
Pascal Compiler v4	£146
Quick BASIC v4.5	£65
Quick C v2.0	£65
Quick C & Assemb v2.01	£102
Quick PASCAL	£65
Windows 286 v2.1	£65
Windows 386 v2.1	£119
Excell v2.1	£287
Multiplan v4.01	£119
Windows Draw v1.01	£115
Windows Software Dev	£299
LOTUS DEVELOPMENT	
Freelance Plus v2.01	£269
Freelance Maps	£95
Freelance Plus v3.01	£269
Lotus 123 Rel 2.2	£269
Lotus 123 Rel 3.0	£319
Lotus Symphony v2.2	£425
Agenda v1.01	£221
Magellan	£92

SCO XENIX	286	386
SCO Xenix Op Sys	£357	£422
SCO Xenix Dev Sys	£357	£455
SCO Xenix Text Proc	>	£126
SCO Xenix Comp Sys	£775	£905
SCO Unix Op Sys		£525
SCO Unix Devel Sys		£613
SCO CGI Graph Run/Sys	>	£191
SCO Multiview	£256	£321
SCO VP/ix		£321
SCO Foxbase+	£516	£647
SCO Foxbase+ Run/Sys	>	£191
SCO Masterplan	>	£516
Multiplan	>	£321
SCO Eroff Documentor	£451	£516
SCO Lyrix v6.0	>	£386
SCO Professional	>	£386
SCO Image Builder	>	£386
M/soft Word v5.0	>	£563
M/soft BASIC compiler	>	£451
M/soft BASIC interp.	>	£256
M/soft FORTRAN comp.	>	£451
M/soft PASCAL comp.	>	£451
SCO XENIX-NET Server	£386	£451
SCO XENIX-NET 3 Server	£841	£1036
SCO TCP/IP (Req streams) Run/Sys		£227
SCO TCP/IP (Req streams) Dev/Sys		£146
CALL FOR INFORMIX PRICES		

Microport UNIX	286	386
Complete System	£395	£495
Comp. Sys. Unlimited	£538	£718
Runtime System (2 user)	£159	£189
Run. Sys. Unlimited	£299	£358
Unlimited User Upgrade	£159	£189
Software Development	£189	£329
Text Preperation	£129	£129
Network Services Extension (Includes Streams & RFS)		£119
DOS Merge Limited	£165	£249
DOS Merge Unlimited	£165	£319
(Requires relevant Runtime Sys)		
Informix		
4GL Devel	£897	£1497
4GL Runtimes	£450	£750
SQL Devel	£717	£837
SQL Runtime	£357	£417
ESQL/C Devel	£537	£657
ESQL/C Runtime	£270	£330
C-ISAM	£199	£229
Individual Manuals Each.	£60	£60
User's Guide	£60	
Advanced Programming Guide	£60	
Microport Sys V Made Easy	£25	

INTERACTIVE UNIX 386/ix	
Application Starter (1-2 Users)	£485
Application Extender (1-2 Users)	£645
Application Developer (1-2 Users)	£925
Network Starter (1-2 Users)	£365
Network Extender (1-2 Users)	£585
Network Developer (1-2 Users)	£965
Workstation Starter (1-2 Users)	£325
Workstation Extender (1-2 Users)	£715
Workstation Developer (1-2 Users)	£1285
Software Developement System	£485
NFS with TCP/IP (1-2 Users)	£435
X-Windows (Runtime)	£225
X-Windows (X11.3 Development)	£560

UNIX Servers for ALL Budgets	
Starter KIT: £1195	Corporate KIT: £1895
286/12 w 1.5M RAM, 1.2M floppy, 101 keyboard, 3 I/O ports, 20M HD, mono screen. + loadedwith Microport UNIX and DOS.	386/16 w 2M RAM, 1.2M floppy, 101 keyboard, 3 I/O ports 40M HD, mono screen. + loaded with Interative UNIX, VP/IX, TEN/Plus and DOS.

TCP/IP

ETHERNET

XENIX NET

NOVELL

INTERACTIVE 386 IX

MOTIF

OS/2

UNIX

TOKEN RING

PCNX
£129.00

DOS

ESIX/V 386
£313.00

MINIX
£89.00

AIX

SCO UNIX

INTEL 386 IX

Open Desktop

Centre Point Bargains - 1 minute from Tottenham Court Road tube

These items £50 EACH

Essential B Tree - C Doc - PC Expert - M2C2Basic - CPM86 Toolbox - ZView - OS/2 Standard Edition Ver 1.0 - Screenstar - Canon Photocopier (Used).

These items a little more BUT still a BARGAIN!

Raima dbFile (source)	£275
Raima dbRetrieve (source)	£275
CB Tree	£95
Essential Graphics (source)	£295
C Utility Library	£75
Panel Plus (source editor)	£750
Displaywrite 4/2	£75
Terminal Control (source)	£595
Vitamin C (source)	£149

Here's the HARDWARE

PC Brands (IBM Comp. - All NEW)

286/12	from	£529
386SX-16	from	£899
386/20	from	£1259
386/25	from	£1499
386/33	from	£2299

IBM Worms Drive - (EX-DEMO) 200/cart (RRP: £2395)	£1495
IBM Quietwriter III (EX-DEMO)	£395
Qume Scripten (EX-DEMO) Postscript Printer	£2650
Compaq 386/20 Portable (EX-DEMO) 6M RAM, 100M HD	£4950

Don't forget these !!

Complete Systems.

SCO XENIX 286 £775
SCO XENIX 386 £905

UNIX

MICROPORT SYS V 286 £395
MICROPORT SYS V 386 £495
Complete Systems.

MKS Toolkit (Unix Like)
£145

System IX Networks Ltd

16 Bedford Court Mansions
Bedford Avenue,
London WC1B 3AD
Fax: 071 - 255 1038

Orders/Sales - (6 lines)

071-636 8210

Customer Services: 071 - 631 4816

Compaq | Toshiba | IBM | Epson | Qume | NEC | Seagate | Wyse | Relysis | Zenith | Priam | Ranger

CIRCLE NO. 912

X-BUFFER FROM MICRO CONTROL SYSTEMS FOR ALL THOSE WHO'VE GOT BETTER THINGS TO DO WITH THEIR PC THAN WAIT FOR THEIR PRINTER

micro
control systems

Imagine you've got a 30 page report on your PC. The average printer will take about 17 minutes to print out, during which time your PC is either completely incapacitated or operating far slower than usual.

So,what can you do?

With the MCS X-Buffer, lots.

WHY THE PROBLEM?

A printer operates at a slower rate than a computer, forcing the PC to wait whilst the printing job is completed. And with a plotter producing a complex CAD schematic, that could be several *hours,* not minutes. Fine if you've got time to waste—but most of us haven't.

WHAT IS THE X-BUFFER?

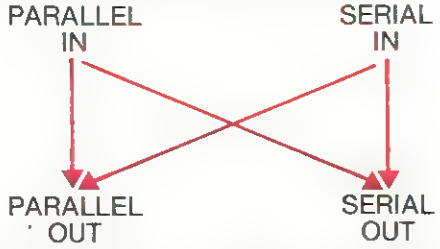
X-Buffer is an intermediate storage device that acts as an extension to the computer's own memory. It allows text or data to be downloaded in seconds and 'held' in the buffer ready to be fed to the printer at the correct rate.

So there's no waiting for printing to finish, the PC is free to be used for other tasks.

In the case of your 30 page report, downloading to the X-Buffer would take around 28 seconds, leaving you 16½ minutes to get on with something far more constructive.

VERSATILITY

Until recently printer buffers had to be specified according to the type of PC and the printer used and the interfaces each had. The X-Buffer is designed for maximum flexibility and incorporates four interfaces: RS232 Serial i/o and Centronics parellel i/o.



The interfaces can be used simultaneously so the buffer can be used in a variety of ways. Two computers sharing one output device (printer, plotter, modem). One computer driving two devices. Two computers using two devices.

EXPANDABILITY

The X-Buffer is also easily expandable by the user, plug-in memory modules offering from 64K to 4 Mbytes of buffer memory.

That's not only good news for users but great news for dealers too—instead of holding large stocks of varying fixed capacity buffer units,the X-Buffer modules allow custom configuration depending on the users' needs.

EXTRAORDINARY VALUE

And the price of the X-Buffer is probably the best feature.

X buffer

FROM **£159**

The 64K basic unit costs a mere £159 RRP including a standard two year parts and labour warranty.

X-Buffer from Micro Control Systems. Don't let your printer (or plotter or modem) keep you waiting.

The X-Buffer is just one of a range of buffer products manufactured and supported in the U.K., available from Micro Control Systems.
Tel: 0602 391 204

AVAILABLE FROM

BYTECH peripherals
Tel: 0344 427676

Northamber 081 974 1711

STC ELECTRONIC SERVICES
Tel: 0279 444445

CIRCLE NO. 913

Hardware Focus

Our regular look at recent hardware launches, as they affect software developers. This month, networks and PS/2s dominate.

PS/2 Range Expands

IBM expanded its PS/2 range at the end of March with the launch of six new computers and a collection of peripherals.

There are four new Model 80s, and two versions of the new Model 65. All the new machines are Micro Channel based, and have SCSI hard disks. It's the first time that IBM's PCs have had SCSI interfaces as standard.

Also announced was a new 14 inch colour monitor.

The Model 80/A31 has a 25 MHz 386, four 32-bit slots and four 16-bit slots. There is a 64 KB cache, 4 MB of RAM as standard (expandable to 8 MB on board, or 16 MB in total), and a 320 MB SCSI hard disk with an access time of 12.5 ms. The Model 80/A21 is the same, but with a 120 MB SCSI disk. The SCSI controller can support up to four SCSI hard disks. Prices are £7831 and £6267.

The Model 80/M21 is a 20 MHz 386 machine, with three 32-bit slots and five 16-bit slots. With a 320 MB SCSI disk it costs £6978. The Model 80/X21 is the same, but with a 120 MB SCSI drive, and costs £5271.

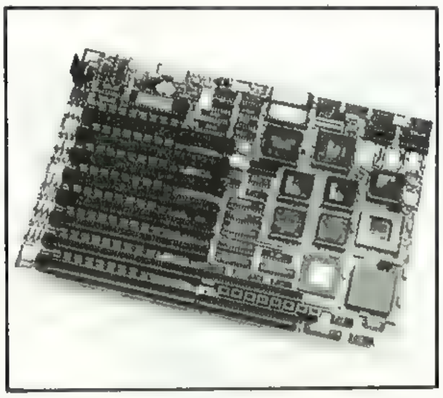
The Model 65/SX 121 has three 32-bit slots, and five 16-bit slots, and runs a 386SX at 16 MHz. It has 2 MB of RAM and a 120 MB SCSI drive, and costs £4101. The 65/SX 061 is the same, but with a 60 MB SCSI hard disk, and costs £3545.

A PS/2 MCA SCSI card, available separately for other PS/2 machines, costs £363 for a 16-bit verison, or £717 for the 32-bit version with cache. The SCSI drives (320 MB, 120 MB and 60 MB) are £3890, £1367, and £854.

IBM has also entered the world of CD-ROM. The internal version for a PS/2 costs £893, and you'll also need an £89 adaptor kit. An external drive costs £1069. This, we can't help noticing, is around five times the price of similar drives that we've seen.

Finally in this batch of announcements, there's an entry level Model 70, a new laser printer with optional PostScript capability and the new 8515 colour monitor.

New Motherboards



Monolithic Systems Corporation is an American hardware manufacturer. According to the dictionary, monoliths are massive and unmoveable, which makes it all the more surprising that MSC have produced a couple of neat 25 MHz 386 motherboards for those who want to build or distribute PCs. The 386CT is the baby-AT version, with eight slots, while the 386CX is the grown-up version with 14. The boards use a Chips & Technologies chipset, and an AMI BIOS. Prices are £1100 for the CT with 2 MB of RAM, and £1370 for a similarly configured CX. A 25 MHz 486 board is also available now. This has eight slots, and costs £2100 when populated with 2 MB. UK distributor is Micro Marketing, which sounds a much more suitable name, who are on 06285 29222.

Instant Networks

CompuAdd is about to launch what the Company likes to call an 'off-the-shelf' network. Bristol-based CompuAdd are putting the finishing touches to 4-user and 8-user network starter packs, aimed at small companies and software development teams. Each kit comprises Novell software and Ethernet hardware, and a CompuAdd 286 or 386-based server. The price, which has not yet been announced, will include cabling and testing. Installation and training services are also available. Installing a single PC is hard enough; installing four of them, and networking them together, is often too complex for end users. CompuAdd's solutions sound like an ideal way to ease the load of a consultant, who can now offload the purchasing and installation to the client, and concentrate instead on configuring the system and writing bespoke software.

LCD VGA

If your desk is so cluttered that you've had to buy a special wheel to exercise your mouse, or you suffer from eye strain when staring at a standard monitor, then you may be interested in a possible solution. It's a high contrast, black and white VGA monitor, based on LCD rather than a CRT. It's 6 cm thick, weighs two kilos and costs £695. Not cheap, but rather good looking. UK distributor, in case you want to call them and ask what the keyboard in the photo is for, are Sygnos Technologies. They can be reached on 01 352 1478.

EXE

The Code Page

Supporting Dual Monitors

This month's Code Page explains how to connect two monitors to a PC simultaneously, and send different data to each.

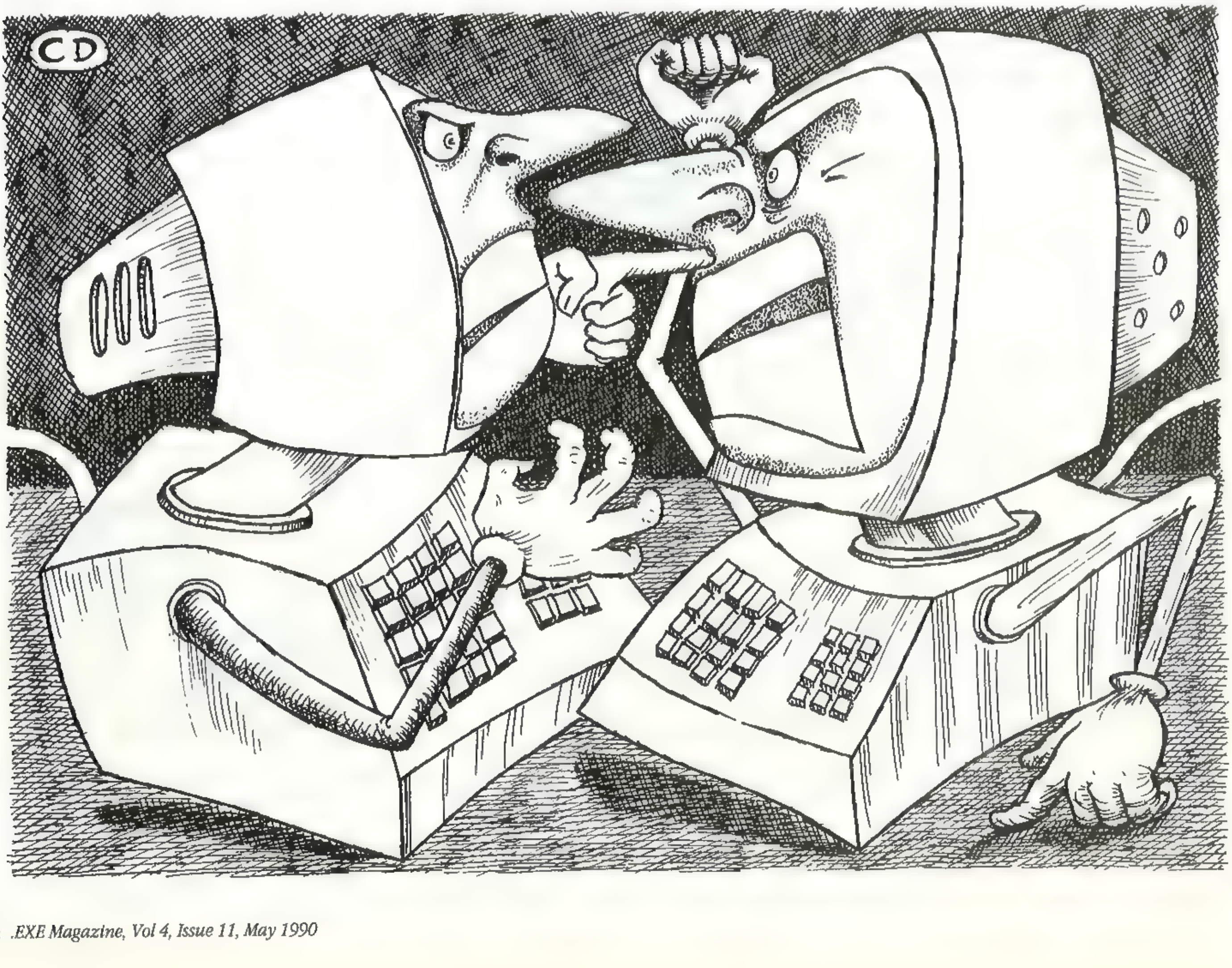
Before we start, an apology. In the .EXE office, we tend to use the OPTASM assembler, because it's faster than MASM. Also, it has the amazingly useful feature of allowing conditional jumps that are out of range. At assembly time, it translates these jumps into code that the CPU can accept. In the past, OPTASM code has crept into the Code Page listings. So, if you've tried to assemble programs, only to have MASM say that conditional jumps are out of range, the necessary corrections should be obvious. If they're not, call the office.

Now to dual monitor support under MS-DOS - the subject of this month's Code Page feature. Many thanks to Bob Stimpson for the contribution.

When IBM decided that the PC should be capable of supporting more than one installed monitor by making the video memory start addresses generally unique, it gave the machine an ability that few have fully exploited. Autocad and CodeView are two current examples from the minority of major vendor software which has used this capability to the full. To the vast majority of programs, however, this remains uncharted territory. For debugging of programs, especially those that normally run in a

```
DECLARE FUNCTION MDAHIBYTE% (Value AS INTEGER)
DECLARE FUNCTION MDALOBYTE% (Value AS INTEGER)
DECLARE FUNCTION MDATWOBYTE% (HiValue AS INTEGER, LoValue AS INTEGER)
DECLARE FUNCTION MDACURROWCOL% ()
DECLARE FUNCTION MDAOK% ()
DECLARE FUNCTION MDASCREEN% (Row AS INTEGER, Col AS INTEGER, Mode AS INTEGER)
DECLARE SUB MDALOCATE (Row AS INTEGER, Col AS INTEGER)
DECLARE SUB MDACURSIZE (StVal AS INTEGER, EndVal AS INTEGER, Mode AS INTEGER)
DECLARE SUB MDAPALETTE (Mode AS INTEGER)
DECLARE SUB MDAPRINT (Printed AS STRING, Attribute AS INTEGER)
DECLARE SUB MDASCROLL (StLine AS INTEGER, EndLine AS INTEGER, Mode AS INTEGER)
```

Declarations to be included in the calling BASIC program



EVERLOCK
COPY PROTECTION

Listing of MDASETB.ASM

```
; MDASETB.ASM contains source code
; for the assembler routines called
; by the MDA access routines in
; QuickBASIC.  MASM v5.0 syntax has been
; used for directives.


; These routines are intended to
; demonstrate the ease with which the
; MDA cán be accessed from a language
; ostensibly only able to support one
; monitor.


; For these routines to run correctly,
; the PC must be set up with the colour
; monitor as the default on boot-up and
; the quick language invoked using this
; colour monitor. For environment work,
; it is recommended that MDASETB.ASM be
; compiled into a Quick Library and the
; /L [libraryname] command line switch
; is used to load it.
; Eg, QB /L MDASETB.QLB.


.MODEL MEDIUM
.CODE


PUBLIC MDAHIBYTE
PUBLIC MDALOBYTE
PUBLIC MDATWOBYTE
PUBLIC MDACURROWCOL   ;Returns csr col/row
PUBLIC MDACURSIZE     ;Sets or hides crsr
PUBLIC MDALOCATE      ;Positions cursor
PUBLIC MDAOK          ;Finds MDA card
PUBLIC MDAPALETTE     ;BLINK control
PUBLIC MDAPRINT       ;Prints to MDA screen
PUBLIC MDASCREEN      ;Returns char or attr
PUBLIC MDASCROLL      ;Scrolls or clears MDA
                      ;screen


MDAHIBYTE  PROC
PUSH   BP          ;set up stack frame pointer
MOV    BP,SP
MOV    BX,[BP+6] ;get BASIC argument address
MOV    AX,[BX]   ;load into AX
MOV    AL,AH     ;put high byte in AL
XOR    AH,AH     ;clear AH
POP    BP        ;restore stack
RET    2         ;exit
MDAHIBYTE  ENDP


MDALOBYTE  PROC
PUSH   BP          ;set up stack frame pointer
OV    BP,SP
MOV    BX,[BP+6] ;get BASIC argument address
MOV    AX,[BX]   ;load ito AX
XOR    AH,AH     ;clear AH
POP    BP        ;restore stack
RET    2         ;exit
MDALOBYTE  ENDP


MDATWOBYTE PROC
PUSH   BP          ;set up stack frame
MOV    BP,SP
MOV    BX,[BP+8] ;get BASIC HIGH BYTE
                 ;argument address
MOV    DX,[BX]   ;load value into DX
MOV    BX,[BP+6] ;get BASIC LOW BYTE
                 ;argument address
MOV    AX,[BX]   ;load value into AX
MOV    AH,DL     ;load high byte into AH
POP    BP        ;restore stack
RET    4         ;exit
MDATWOBYTE ENDP


MDACURROWCOL  PROC
MOV    DX,3B4h   ;select MDA 6845 CRTC
                 ;address port 3B4
MOV    AL,0Eh    ;address register 0E
                 ;(cursor high byte)
OUT    DX,AL     ;activate CRTC
INC    DX        ;select CRTC data port 3B5
IN     AL,DX     ;get cursor high byte
MOV    AH,AL     ;move high byte to AH
DEC    DX        ;select CRTC address port 3B4
MOV    AL,0Fh    ;address register 0F
                 ;(cursor low byte)
OUT    DX,AL     ;activate CRTC
INC    DX        ;select data port
IN     AL,DX     ;get cursor low byte
                 ;AX now contains cursor pos
MOV    BL,50h    ;load characters per row (80)
DIV    BL        ;divide AX by BL:
                 ;cols now in AL, rows in AH
XCHG   AH,AL
INC    AH        ;inc, to use BASIC offset
INC    AL        ;instead of CRTC zero offset
RET              ;return with data in AX
MDACURROWCOL  ENDP


MDACURSIZE    PROC
PUSH   BP
MOV    BP,SP     ;set up stack addressability
MOV    BX,[BP+6] ;get Size control
MOV    AX,[BX]   ;put value in AX
OR     AX,AX     ;is it = 0?
JZ     SHORT HideSize  ;hide the cursor
MOV    BX,[BP+10]      ;get START variable
                       ; address from stack
MOV    AX,[BX]         ;put variable in DX


CMP    AX,0            ;row less than zero?
JB     SHORT CursizeExit  ;error- so exit
CMP    AX,13              ;is start row > 13?
JA     SHORT CursizeExit  ;error -so exit
MOV    CL,AL              ;store start in CL
MOV    BX,[BP+8]          ;get END variable
                          ;address from stack
MOV    AX,[BX]            ;put variable in AX
CMP    AX,0               ;row less than zero?
JB     SHORT CursizeExit  ;error- so exit
CMP    AX,13              ;row more than 13?
JA     SHORT CursizeExit  ;error -so exit
MOV    CH,AL              ;store end in CH


Setsize:


MOV    DX,03B4h  ;load MDA address reg value
MOV    AL,0Ah
OUT    DX,AL     ;activate CRTC
INC    DX        ;address CRTC data register
MOV    AL,CL     ;load low cursor size byte
OUT    DX,AL     ;send to CRTC
DEC    DX        ;load MDA address port value
MOV    AL,0Bh    ;load size high
OUT    DX,AL     ;activate CRTC
INC    DX        ;address CRTC data register
MOV    AL,CH     ;load high cursor size byte
OUT    DX,AL     ;send to CRTC

CursizeExit:


POP    BP        ;clean up and exit
RET    6


Hidesize:


MOV    CL,20h    ;set invisible
                 ;positions for start
MOV    CH,20h    ;and end cursor size
                 ;locations
JMP    Setsize   ;now drive the CRTC
MDACURSIZE   ENDP


MDALOCATE    PROC
PUSH   BP        ;set up stack pointer
                 ;addressability
MOV    BP,SP
MOV    BX,[BP+8] ;get addr of ROW value
MOV    AX,[BX]   ;move row count into AX
DEC    AX        ;correct for zero offset
CMP    AX,0      ;check for valid row
JB     SHORT ExitLocate  ;if below 0,exit
CMP    AX,24
JA     SHORT ExitLocate  ;if above 24 exit
MOV    BL,80     ;set up for chars per row
MUL    BL        ;multiply AX by BL
MOV    CX,AX     ;save result in CX
MOV    BX,[BP+6] ;get address of COL value
                 ;in BASIC
MOV    AX,[BX]   ;move col number into AX
DEC    AX        ;correct for zero offset
CMP    AX,0      ;check for valid col
JB     SHORT ExitLocate  ;if below 0, exit
CMP    AX,79
JA     SHORT ExitLocate  ;if above 79, exit
ADD    CX,AX     ;add columns,row total for
                 ;full offset
MOV    DX,3B4h   ;select CRTC address
MOV    AL,0Eh    ;address cursor high byte
OUT    DX,AL     ;activate MDA CRTC
INC    DX        ;address data port 3B5
MOV    AL,CH     ;put high byte cur pos in AL
OUT    DX,AL     ;send to CRTC
DEC    DX        ;select address port 3B4
MOV    AL,0FH    ;address cursor low byte
OUT    DX,AL     ;activate MDA CRTC
INC    DX        ;select data port address 3B5
MOV    AL,CL     ;low byte of cur pos in AL
OUT    DX,AL     ;send to CRTC


ExitLocate:


POP    BP        ;restore segment pointer
RET    4         ;return with 4 bytes of args
MDALOCATE   ENDP



MDAOK       PROC
MOV    DX,03B4h  ;load MDA address reg value
MOV    AL,0Eh    ;load cursor high reg value
OUT    DX,AL     ;activate CRTC
INC    DX        ;address CRTC data register
IN     AL,DX     ;get current high byte value
MOV    AH,AL     ;save in AH
MOV    AL,37h    ;set AL to arbitrary value


OUT    DX,AL     ;send to CRTC
JMP    MdaReady1 ;wait for CRTC to respond
```

graphics environment such as Windows, twin monitors are essential - a colour monitor runs the graphics environment, while a mono monitor displays debugging information.

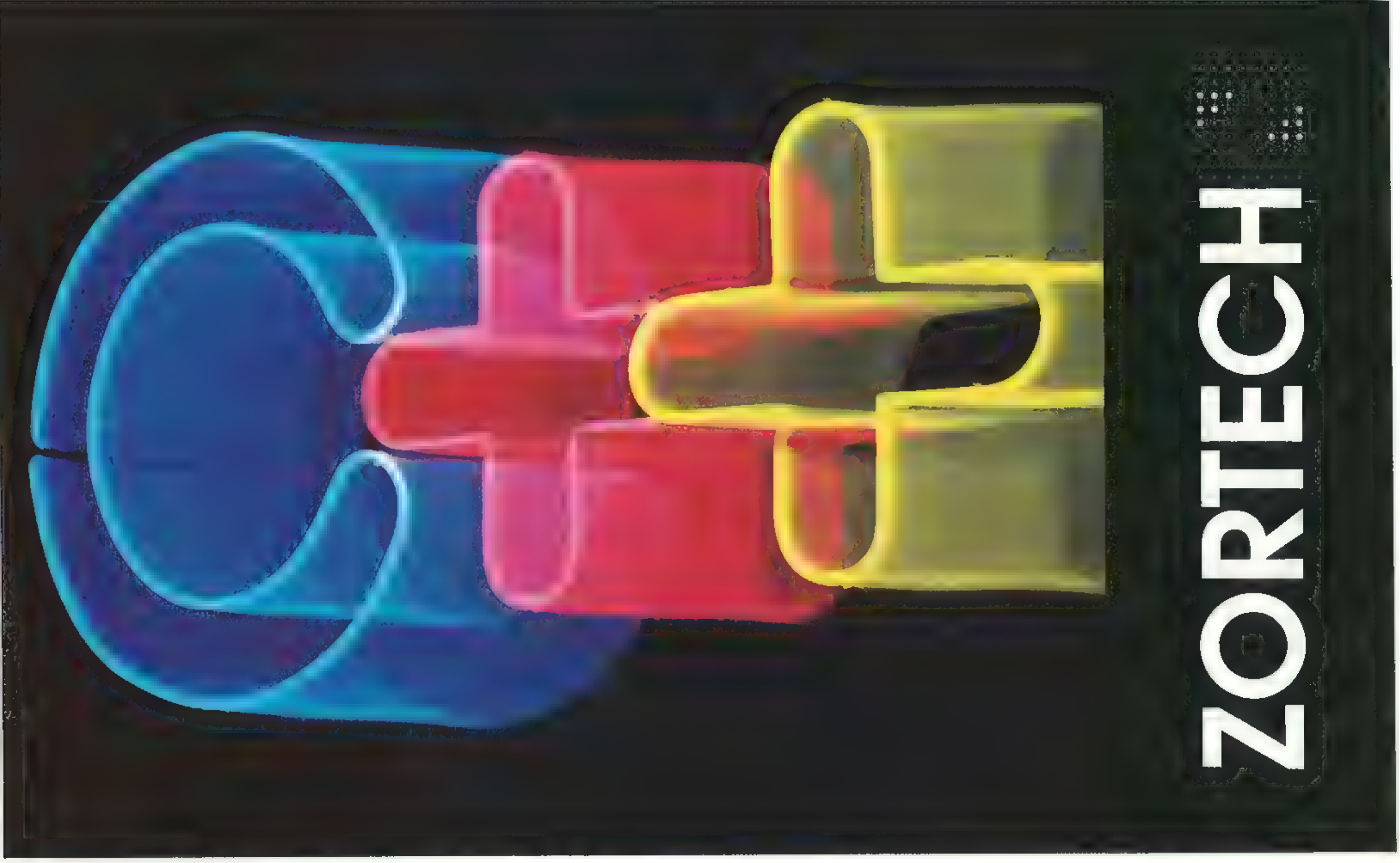
It could be said that there are not many users currently with operational dual-monitor systems or hardware - and until quite recently this was probably correct. There is an accelerating acceptance of VGA and EGA displays for the now standard 80286 and 80386 based machines. As these drop in price and existing users upgrade to better monitors, there will an increasing number of surplus MDA monitors widely available. These are currently so cheap, that it's well worth getting hold of one.

Two displays double the amount of data presented to the user - so it is the program designer's increasing responsibility to ensure the system is presented in a consistent and ergonomic manner. It becomes very easy to create twice as much junk display output with two monitors, and this may easily succeed in confusing the user to a far greater extent than simply by a factor of two! This needs to be borne in mind if you are thinking of writing some dual-display software.

Tools

The object of this article is to provide the reader with tools to evaluate the benefits of dual-monitor systems. Writing directly to video memory is one solution to the problem of addressing two monitors, but it must be done using assembler, otherwise it is grossly inefficient and the screen changes will be slow and discernable. In considering how to approach the overall problem, I decided that the simplest approach was the best - leave the graphics/colour monitor control to the experts who produced my Microsoft language compiler, and I would provide a library to control the MDA.

The routines presented here interface directly with Microsoft QuickBASIC or BASIC 6.0. Thus the assembly language interface expects arguments passed by near reference. The BASIC strings must be stored in DGROUP (the default) for these demo routines to function - hence the requirement for only the offset to the start of the string and not also the segment. Strings are passed by reference to a BASIC string descriptor (four bytes). The first word of the descriptor is a two-byte integer describing the length of the string, and the second word is the near offset to the start of the string. For C programmers this should not prove a problem - the routines can be easily modified to suit your own preferences or calling mechanism if you need wider versatility.



"Zortech C++ is one of the best MS-DOS products I've had the luck to use. I can highly recommend Zortech V2.0"

Scott Robert Ladd – Dr. Dobbs Journal – January 1990

NEW! AT&T C++ V2.0 SPECIFICATION

NEW! MS WINDOWS COMPATIBILITY

NEW! C++ SOURCE LEVEL DEBUGGER

NEW! EXPANDED C++ TOOLS

NEW! OS/2 COMPILER UPGRADE

NEW! EASIER PORTABILITY FROM MSC

Zortech is first again with the release of its new C++ V2.0 Developer's Edition featuring the AT&T C++ V2.0 specification.

New V2.0 features like Multiple Inheritance and Type Safe Linkage make this the most advanced compiler available today.

You get 1500 pages of clear, high quality, professional documentation.

Zortech C++ V2.0 makes it really easy for you to move code over from most other leading C compilers.

Zortech C++ V2.0 Developer's Edition comes with a great new environment that lets you edit, compile and debug with ease.

Zortech present another "World's First" with its new C++ Source Level Debugger for MS-DOS. Once you've used our debugger you will never want to go back to any other.

The Developer's Edition also includes a 99% ANSI compatible C compiler, seamless LIM/EMS support, C++ Graphics Shell, TSR functions, C++ Tools, Optimizer, SAA/CUA style user interface, and full standard library source code.

Please call for our color brochure.

PRICES

C++ Compiler	£129.95
C++ Debugger	£99.95
C++ Tools	£99.95
Library Source	£99.95

Save £130 – Get the Developer's Edition for only £299.95 (includes all the above items).

OS/2 Compiler	£CALL
C++ Video	£299.95

USA: Zortech Inc.
1165 Massachusetts Ave.
ARLINGTON MA02174
Voice: 617-646-6703
Fax: 617-643-7969

EUROPE: Zortect Ltd.
106-108 Powis Street
LONDON SE18 6LU
Voice: 44-1-316-7777
Fax: 44-1-316-4138

HOTLINE 01-316-7777

CIRCLE NO. 918

Listing of MDASETB.ASM continued

```
MdaReady1:


JMP     MdaReady2 ;(these take 20 or 30 clocks,
                  ;depending on CPU)


MdaReady2:


IN      AL,DX      ;load current setting
XCHG    AH,AL      ;place original in AL
OUT     DX,AL      ;restore original value
MOV     BL,AH      ;save in BL
XOR     AX,AX      ;clear AX for return
                   ;MDAOK = false
CMP     BL,37H     ;are set and read
                   ;values the same?
JNE     SHORT BadMDA ;no - return with AX = 0
MOV     AX,1          ;yes an MDA - so
                      ;return with AX = 1


BadMDA:


RET
MDAOK     ENDP


MDAPALETTE     PROC
PUSH    BP  ;set up stack pointer
            ;addressability
MOV     BP,SP
MOV     BX,[BP+6]  ;get address of PALETTE
MOV     AX,[BX]    ;move into AX
CMP     AX,1       ;if palette 1, set the
                   ;blink bit
JE      SHORT SetBlink
CMP     AX,0
JNE     SHORT SkipPalette ;error - so exit
MOV     DX,3B8h        ;disable blink
                       ;address CRTC register
MOV     AL,00001000b ;blink bit(5) in AL
                     ;control byte = 0
OUT     DX,AL          ;send to CRTC
JMP     SHORT SkipPalette  ;error


SetBlink:


MOV     DX,3B8h        ;enable blink
                       ;address CRTC register
MOV     AL,00101000b ;blink bit (5) enabled
OUT     DX,AL          ;send to CRTC


SkipPalette:


POP     BP             ;restore stack pointer
RET     2              ;return
MDAPALETTE  ENDP


MDAPRINT   PROC
PUSH    BP          ;set up stack frame pointer
MOV     BP,SP
PUSH    ES          ;save registers which will be
PUSH    SI          ;changed
MOV     DX,3B4h     ;select MDA 6845 CRTC
                    ;address port 3B4
MOV     AL,0Eh      ;address register 0E
                    ;(cursor high byte)
OUT     DX,AL       ;activate CRTC
INC     DX          ;select CRTC data port 3B5
IN      AL,DX       ;get cursor high byte
MOV     AH,AL       ;move high byte to AH
DEC     DX          ;select CRTC address port 3B4
MOV     AL,0Fh      ;address register 0F
                    ;(cursor low byte)
OUT     DX,AL       ;activate CRTC
INC     DX          ;select data port
IN      AL,DX       ;get cursor low byte
                    ;AX now contains pos
SHL     AX,1        ;double to get byte offset
MOV     DI,AX       ;load as destination offset
MOV     BX,[BP+8] ;get string descriptor addr
MOV     CX,[BX]   ;load string length into CX
JCXZ    ErrorExit ;if it is a null string,exit
MOV     SI,[BX+2] ;load offset of string
MOV     AX,0B000h ;load MDA segment into
MOV     ES,AX     ;ES for addressing
                  ;the video memory
MOV     BX,[BP+6] ;address of attribute byte
MOV     AX,[BX]   ;load value into AX
MOV     AH,AL     ;put attribute byte into AH


PrintString:


MOV     AL,[SI]       ;place string char into AL
INC     SI            ;SI to next char.
MOV     ES:[DI],AX    ;move char and attr to
                      ;MDA video mem
ADD     DI,2          ;inc to next char posn
LOOP    PrintString ;transfer until CX = 0
```

```
ErrorExit:


POP     SI             ;restore registers
POP     ES


POP     BP
RET     4
MDAPRINT    ENDP


MDASCREEN   PROC
PUSH    BP
MOV     BP,SP
MOV     BX,[BP+10]          ;get address of ROW
MOV     AX,[BX]             ;move row count into AX
DEC     AX
CMP     AX,0                ;check for valid row
JB      SHORT ExitScrn      ;if below 0,exit
CMP     AX,24
JA      SHORT ExitScrn      ;if above 24 exit
MOV     BL,160              ;bytes per row
MUL     BL                  ;calc word offset
MOV     CX,AX               ;save result in CX
MOV     BX,[BP+8]           ;get address of COL
MOV     AX,[BX]             ;col number into AX
DEC     AX
CMP     AX,0                ;check for valid column
JB      SHORT ExitScrn      ;if below 0, exit
CMP     AX,79
JA      SHORT ExitScrn      ;if above 79, exit
SHL     AX,1                ;get byte offset
ADD     CX,AX               ;full offsot
MOV     BX,[BP+6]           ;get address of Mode
MOV     DX,[BX]             ;load value into DX
PUSH    DS
PUSH    SI
MOV     AX,0B000h
MOV     DS,AX               ;MDA seg addr
MOV     AX,CX               ;load SI with offset
MOV     SI,AX
PUSHF                       ;save flags
CLD
LODSW                       ;DS:[SI] into AX
POPF
POP     SI
POP     DS
CMP     DL,0                ;if 0, then (AL) wanted
JE      SHORT ExitScrn      ;so don't move
                            ;character value
MOV     AL,AH               ;it's not 0, so
                            ;attribute wanted


ExitScrn:


XOR     AH,AH               ;AH not needed
POP     BP
RET     6                   ;return 6 bytes of args
MDASCREEN      ENDP


MDASCROLL      PROC


PUSH    BP               ;set up stack pointer
                         ;addressability
MOV     BP,SP
MOV     BX,[BP+10]
MOV     DX,[BX]          ;put variable in DX
DEC     DX               ;correct for zero offset
CMP     DX,0             ;start row less than zero?
JB      SHORT ScrollExit  ;error- so exit
CMP     DX,23                 ;start more than 23?
JA      SHORT ScrollExit  ;error -so exit
XCHG    DH,DL                 ;store start in DH
MOV     BX,[BP+8]        ;get END variable addr
MOV     AX,[BX]          ;put variable in AX
DEC     AX               ;correct for zero offset
CMP     AL,DH            ;less or equal to start?
JLE     SHORT ScrollExit  ;error - so exit
CMP     AL,24                 ;end row off screen?
JA      SHORT Scrollexit  ;error - so exit
MOV     DL,AL                 ;store end in DL
MOV     BX,[BP+6]             ;scroll ctrl val addr
MOV     AX,[BX]               ;put value in AX
CMP     AX,0                  ;is it = 0?
JE      SHORT ClearAll
CMP     AX,1                  ;is it = 1?
JE      SHORT ScrollUp1   ;scroll up 1
                          ;error, so exit
ScrollExit:


POP     BP                    ;clean up and exit
RET     6


ClearAll:


XOR     AX,AX            ;clear AX to calc clear
                         ;start offset
MOV     AL,DH            ;load AL with start line
MOV     BX,160
MUL     BL               ;start offset in AX
MOV     CX,AX            ;save in CX
XOR     AX,AX
```

Gaining Control

The primary operations needed for effective control of the MDA are: checking/confirming it is installed and initialised; clearing a display viewport on the screen; setting the character display attributes; displaying a data string; returning a character code or attribute at a specific location; scrolling a display viewport up one line; maintaining full control of the cursor size and location.

These functions can be further subdivided down into detailed operations, such as operations to write a string of text with attribute bytes inserted within it, or write a string of plain text using a separate attribute byte. The approach here has been to provide the simple, straightforward, minimum consistent with a working evaluation system. It would not be difficult to add other routines which emulated specific language commands or DOS interrupts if needed.

The structure of the routines and functions presented here has been organised to provide the programmer with the most straightforward method of accessing the MDA from a DOS programming environment. The most frequently used display commands have been emulated to provide an effective method for MDA control.

It is important to note that these demonstration routines contain only the most elementary error checking code - for an operational library, the error trapping must be improved to your own in-house requirements.

Preparation

To use these routines, the assembler routines should be entered and assembled using MASM 5.0 or above to an object file. Using LINK with a command line such as: `LINK /Q MDASETB.OBJ, MDASETB.QLB,, BQLB40.LIB;` will create a Quick library which can be loaded into the QuickBASIC environment when invoking it by including the command `/L MDASETB.QLB` on the QB invoking command line. The QuickBASIC program can then be entered and the assembler routines accessed directly. To compile into a .EXE file from within the environment, you must first produce a library file by invoking LIB with the instructions

`LIB MDASETB.LIB+MDASETB.OBJ;`

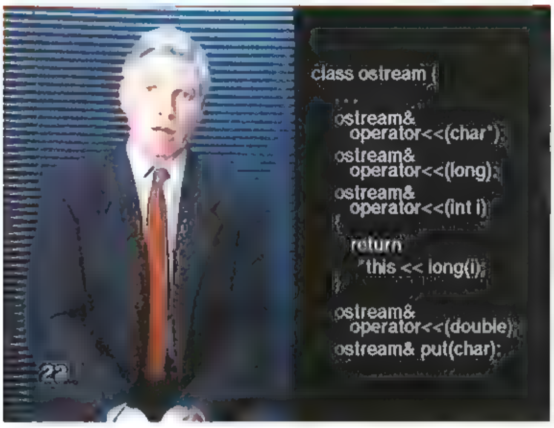
Without this file, the QB compiler will abort stating 'library file not found'.

The Routines in Detail

The presence of an MDA in the system is verified by MDAOK which returns a value

VIDEO COURSE

Learn C++



```
class ostream {
   ...
   ostream&
      operator<<(char*);
   ostream&
      operator<<(long);
   ostream&
      operator<<(int i)
   {
      return
       *this << long(i);
   }
   ostream&
      operator<<(double);
   ostream& put(char);
```

NO RUSH!
NO TRAVEL!
NO HOTELS!
ONLY £299.95

Learn C++ Now!

The great exodus of programmers from C to C++ has begun! Since C++ builds on C, it's the easiest OOP language to learn. That's why it's called "the language of the 90's".

Why the rush?

PRODUCTIVITY! Yes, C++ programmers can write programs in less time requiring less maintenance. Large projects become much easier to manage.

Unfortunately, learning C++ can be very costly. Classroom instruction is expensive even without the travel and hotel costs. Of course, not learning will cost you even more in the long run. Now there is an alternative!

The top C++ video tutorial at the lowest possible price.

The C++ video tutorial from Zortech is the ultimate C++ training tool for work or home at only £299.95. It comes on six VHS video tapes containing 32 clear, extensive tutorials. Used in conjunction with the concise workbook and tutorial disk, you will find everything you need for fast-track C++ tuition.

The course is generic (i.e. compiler and hardware independent) and is available with or without the award winning Zortech C++ Compiler for MS-DOS and OS/2.

Start writing C++ code within a week.

As a C programmer, you will start producing C++ code within a week of concentrated use of this course. Alternatively, spend just an hour a day watching the video and working through the suggested exercises to learn C++ in only six weeks!

The Leader in video tutorials.

If you don't already know C, you can join tens of thousands of programmers world-wide who have learnt C with the Zortech Complete C Video Course, described as:

"An excellent bargain ...
... I heartily recommend"
Gary Ray, PC WEEK

Save your company thousands of £££'s.

One programmer can train for only £299.95, but you can train ten programmers for just:

C++ Course	*£299.95*
9 Extra Workbooks	*£179.55*
Total 10 students	*£479.50*

Yes! Only £47.95 each! (With all the FREE refresher courses you need!)

Now, Zortech with its new C++ Video Tutorial has refined the art of video tuition and presentation even further for the 1990's.

Only £299.95 complete

- *Six Videos with 32 lessons*
- *256 page workbook*
- *Tutorial disk*
- *Compiler & hardware independent*
- *NTSC or PAL format*
- *Tax deductible*

Don't delay, order now!

Just mail the coupon or call the order hotline for same day shipment.

USA: Zortech Inc.,
1165 Massachusetts Avenue,
ARLINGTON, MA02174
Voice: 617-646-6703
Fax: 617-643-7969

EUROPE: Zortech Ltd.,
106-108 Powis Street, LONDON
SE18 6LU
Voice: 44- 1-316-7777
Fax: 44- 1-316-4138

IN EUROPE CALL:

44- 1-316-7777

ORDER FORM

Please rush me these items:

Qty	*Description*	*Price*
___	*C++ Video Tutorial*	*£299.95*
___	*Extra C++ Workbooks*	*£. 19.95*
___	*Zortech C++ Compiler*	*£129.95*
___	*Developer's Edition*	*£299.95*

EXE 3/90

Please add £5 + VAT postage in UK
All UK orders please add 15% VAT
International shipping charged at cost

Name ____________________
Company ____________________
Address ____________________
____
____
Phone ____________________
MC,VISA or CHQ ________ *Expiry* ______
Card No. ____________________

CIRCLE NO. 919

Listing of MDASETB.ASM contined

```
MOV     AL,DL           ;load AL with end line
SUB     AL,DH           ;subtract start line
INC     AL              ,get true size in lines
MOV     BX,80           ;number of chars per row
MUL     BL              ;multiply to get size

                        ;of area in AX
MOV     BX,CX           ;put offset in BX
MOV     CX,AX           ;CX as number of words

ClearBlock:

PUSH    DI              ;save ES:DI register pair
PUSH    ES
MOV     AX,0B000h       ;MDA segment value into ES
MOV     ES,AX
MOV     DI,BX           ;get offset from BX
PUSHF                   ;save flag status
CLD                     ;clear direction flag
MOV     AX,0007h        ;load AX with null char
CLI                     ;halt interrupts
REP     STOSW           ;move word in AX to ES:DI
                        ;CX times, incrementing DI
STI                     ;restore interrupts
POPF
POP     ES              ;restore register values
POP     DI
JMP     ScrollExit      ;finished - so exit

ScrollUp1:

XOR     AX,AX
MOV     AL,DH
INC     AL
MOV     BX,80           ;mult by 80 chars/row
MUL     BL              ;start offset in AX
SHL     AX,1            ;get byte offset
MOV     CX,AX           ;move to CX
XOR     AX,AX
```

```
MOV     AL,DL           ;load AL with  end line
SUB     AL,DH           ;subtract start line
MOV     BX,80           ;multiply to get
                        ;size of scrn area AX
MUL     BL
MOV     BX,CX           ;save offset in BX
MOV     CX,AX           ;load in CX as words
PUSHF                   ;save flag status
PUSH    DI              ;save ES:DI values
PUSH    SI              ;and DS:SI
PUSH    ES
PUSH    DS
MOV     AX,0B000h       ;put MDA segment B000
MOV     ES,AX           ;into ES
MOV     DS,AX           ;and DS
MOV     AX,BX           ;get start offset from BX
MOV     SI,AX
SUB     AX,160          ;calc port start line
MOV     DI,AX           ;load DI as scroll dest
CLD                     ;clear direction flag
CLI                     ;halt interrupts
REP     MOVSW           ;move word in DS:SI to
                        ;ES:DI for CX times
STI                     ;restore interrupts


POP     DS
POP     ES              ;restore register values
POP     SI
POP     DI
POPF                    ;scroll complete
                        ;bottom port line
XOR     AX,AX           ;clear AX to calc
                        ;offset of base line
MOV     AL,DL           ;lower line value in AL
MOV     BX,80           ;characters per line
MUL     BL              ;mul to get offset in AX
SHL     AX,1            ;get byte offset
MOV     BX,AX           ;save in BX
MOV     CX,80 .         ;number of words on line
JMP     ClearBlock      ;jump out
MDASCROLL       ENDP
                END
```

of one if it finds an MDA and zero if it does not. (Example: `IF MDAOK% THEN PRINT "MDA FOUND"`). `MDAOK` checks for the presence of the MDA by attempting to address one of the two registers on the CRTC.

It first reads the current cursor high byte from register 0Eh. It then sets a new byte (arbitrarily here at 37h) and then reads what it has just set. If the written and read values agree, then it is safe to assume an MDA is present. `MDAOK` takes a total of 100 clock cycles to complete this activity, the cursor being misplaced for 50 of these cycles.

`MDASCROLL` is the most complex of the routines presented here. The first two arguments are the top row and bottom row of a screen window which is to be processed. The contents are scrolled up one line, or cleared, according to the value of the third argument. The scroll is the full width of the screen.

Writing a string to the MDA is handled by `MDAPRINT` which takes as its arguments a BASIC string descriptor and an attribute byte. The string is written to the screen at the current cursor location. The string can be any length (other than a null string). (Example: `MDAPRINT (FRED$,7).`)

The movement of string data from one part of memory to another for scrolling has been achieved using the `REP MOVSW` mnemonic. This may not at first seem sensible - but the only reason why it is not used in other routines published elsewhere is because they have to take into consideration the need to drive a CGA, and overcome the accompanying 'snow' interference problem. This does not exist on the MDA, so I felt quite free to use the fastest possible method available.

For debugging Windows programs, twin monitors are essential

The MDA cursor size is controlled by `MDACURSIZE` which will also conceal the cursor if required. (Example: `MDACURSIZE (startline%, endline%,visibility)` Visibility is one or zero.)

The cursor can be concealed from view by setting the character cursor lines to non-displayable values. The MDA character size is 14 by nine pixels. The most reliable method conceals it by setting both the start and end values for the cursor lines to 20h. Setting just the high byte may have unpredictable results, so your routine should set both bytes. This requires access to a write-only CRTC port. If you want to refer to the value you are writing to this port, save it elsewhere as a variable. DOS does this for the cursor size on the default monitor system by updating the bytes at 00460h and 00461h with the low (start) and high (end) values.

`MDALOCATE` positions the cursor on the screen. (Example: `MDALOCATE X5,Y%`).

`MDACURROWCOL` returns the current row and column positions of the MDA. (Example: `PRINT MDAHIBYTE%(MDACURROWCOL%)` will print the cursor row, while `PRINT MDALOBYTE%(MDACURROWCOL%)` will print the column).

Closely akin to these routines is `MDASCREEN` which returns either the ASCII code or attribute byte for a specified screen character location. (Example: `PRINT CHR$(MDASCREEN%(1, 1, 0))` returns the character at 1,1, and `PRINT HEX$(MDASCREEN%(1, 1 , 1))` returns the attribute.)

The final routine is `MDAPALETTE` which provides two similar sets of text attributes by enabling or disabling the blink bit in the MDA Mode Control register. (Example: `MDAPALETTE 0`, or `MDAPALETTE 1`). When enabled, blinking text can be displayed on the screen. When this bit is zero, blinking is disabled and the attribute value then controls the intensity of the background attribute.

`MDAPALETTE` accesses the CRTC Control Register 03B8h, to enable or disable blinking text. This register is also used by the Hercules mono graphics card, so you must ensure that, if you are using one, the HGC is in text mode before invoking this routine. If a Hercules card is present in your PC for evaluation of these routines, it defaults to MDA mode ('DIAG' in Hercules terminology) on boot-up. If this is not done (details in many Video guides or the Hercules manuals) setting these bytes at zero for an MDA will change the HGC without changing other registers, thus damaging the display physically. The above precaution is, therefore, only applicable if the Hercules graphics card has been in graphics mode before these demonstration routines are invoked.

.EXE

Bob Stimpson is a Chartered Engineer who initially started finding out how things worked by dismantling his father's wristwatch at the age of five. Unfortunately the wrist-watch didn't work again for 'technical' reasons.

WE MAKE THE PIECES FIT!

pcANYWHERE III

Remote Computing at its best:– £120

The easiest way to resolve user problems from afar: Corporate Information Centres, Software Houses and Computer Dealers can set up diagnostic, support and operational systems, including access to Local Area Networks via dial-up and leased lines. The pcANYWHERE III package includes software for both the host and remote sites.

★ Full control of a host pc from a local pc or terminal allows almost any pc software to be used from a remote site.

★ UK version, with support for wide range of available modems (eg Hayes, Amstrad, Dowty, Miracom, Pace, Tricom and Dacom) including V21, V22, V22bis, V32 and V42 standards.

★ Graphics support for CGA, EGA, MCGA, Hercules and VGA.

★ Help service direct from DMA's authorised UK distributor, including special modem configuration service.

★ Call-back and password security.

★ Fast error checked file transfer.

Call now
02406
6365

UK VERSIONS

Even use an Apple Macintosh running **pcMACTERM (£99)** to control a host pc running pcANYWHERE III, transfer files, cut and paste pc screens into Mac applications, print on either pc or MAC printer.

And for remote control across a local area network, **NetOp (£345)** for NETBIOS compatible networks, and **pcMACTERM/Network (£345)** for Appletalk networks.

CHAIRMAN™

Set up your own Electronic Mail, Bulletin Board or Information System quickly and painlessly using

CHAIRMAN...
£210 TO £710

Supports up to 6 simultaneous lines on one pc, or unlimited callers in network version. Security checking, File transfer, Conferencing, On-line Surveys, Binary Mail transfer, Fully customisable even to the extent of running Database applications.

ASCOM™ IV

ASCOM IV
For General purpose communications via modems or direct connection, this powerful menu or command-driven package allows easy use of Electronic Mail systems, access to mini's and mainframes and other on-line services. Dial directory, Powerful script language, selection of file transfer protocols including XMODEM, YMODEM and Kermit. Terminal emulation includes VT220, VT100 and Data General D200, and can even be used as a terminal to pcANYWHERE III . . . **£170**

TO MAKE YOUR CONNECTIONS RELIABLE AND FASTER HAYES SMARTMODEMS ARE THE ANSWER

V-Series 9600 + Quad	**£999**	**2400 V22, V22BIS**	**£499**
V-Series 2400 + Quad	**£649**	**1200 V21, V22, V23**	**£399**
2400 Quad (V21, V22, V23, V22BIS)	**£599**	**1200B (Internal)**	**£349**

Trade and OEM enquiries are welcome.
Standard terms are CWO. Access and VISA cards accepted.
Prices exclude delivery and VAT.

Hayes™ DMA Authorised distributors

pcANYWHERE III, PCMACTERM, PCMACTERM/Network, CHAIRMAN and ASCOM IV are trade marks of Dynamic Microprocessor Associates Inc. Other brand and product names are trademarks of their respective holders.

Telesystems Ltd
3 Wycombe Road, Prestwood, Bucks HP16 0ND. (02406) 6365

CIRCLE NO. 933

Books

This month, two books for PC developers and one for FoxPro programmers.

PC File Formats and Conversions

A long time ago, John Wiley published two books. One was called 'File Formats for Popular PC Applications', and the other was 'More....'. These books explained the file formats used by Lotus 1-2-3, WordStar 3.3 and a number of other, now-obsolete, applications.

Ralf Kussmann's book, unlike the others, is not simply a collection of reference tables and lists. Instead, it's a collection of easily-readable chapters that explain file formats, and how to convert data between similar applications.

The formats which are fully documented are dBASE IV, IBM DCA/RFT, Microsoft SYLK and Microsoft Word 4.0. The explanations of how to transfer data, cover Pagemaker 3.0, dBASE IV, DisplayWrite 4 V2.0, 1-2-3 V2.2, Freelance Plus V2.01, CHART V3.0, Excel V2.0, Multiplan V4.0, Word V4.0 and V5.0 and Ventura V2.0. The explanations don't include full documentation of the file format - instead, there are details of how to use one application to produce data that is readable by others. Often, there's nothing more involved than clicking an option on a menu. Other times, however, there are macros to write, margin settings to check and so on.

The utility programs and macros listed in the book, as well as an SDF-to-ASCII translator written in Turbo Pascal, are all included on a disk, which accompanies the book.

If you're trying to write conversion utilities, then this is probably not the book you're looking for. However, if you want to know how to convert files using the facilities built into the applications you already have, this is going to be handy to have around. If you can't get hold of a copy, you can order one from Computer Manuals who are on 021 706 6000.

Author: Ralf Kussmann
Publisher: Abacus
Price: £32.45
ISBN: 1-55755-059-X
Pages: 290

PC System Programming

The .EXE shelves are bursting with books that give you a guided tour around the insides of your PC. In our time, we've probably criticised them all. Most of them aren't technical enough. Few dare to mention undocumented DOS areas and data structures. Some cover just DOS and ignore the BIOS. Some cover only the BIOS and ignore DOS.

PC System Programming, on the other hand, ignores hardly anything. It's a superb reference and tutorial on how to do low-level fiddling with just about every part of the PC system. There are dozens of sample programs, in a mixture of Turbo Pascal and assembler (C freaks will have to think back to the days before they saw the light). If you can't face typing in the listings, then there's a pair of disks enclosed (for £55, I suppose, there had jolly well better be).

The first chapters deal with software interrupts - what they are, and how to call them from various languages including GW-BASIC (yes, really), Turbo Pascal and C. Then, in no particular order, come discussions of the interrupt controller chip, the clock, the disk controller, the structure of a device driver and the insides of DOS V4.0. Next comes undocumented DOS structures which explains the workings of memory control blocks, the DOS Info Block and others. The DIB is a table of internal DOS lists, and provides access to device drivers, the system file table (which lets you find the names, rather than the handles, of open files) and so on. Did you know that, if you know where to look, you can find the interleave value of your hard disk?

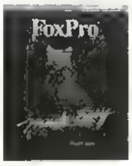
Other chapters include information on determining the amount of free memory, discussions of floppy disks, hard disks, keyboards and printers, writing TSR programs, programming all sorts of video cards and the real time clock, accessing EMS memory, mouse programming and determining the CPU type. There's even a complete DOS and BIOS call summary.

If you are programming MS-DOS PCs at a low or medium level, you'll find this book rapidly becomes indispensable. There's a huge amount of information in it, and you'll be amazed how often you'll find things you didn't know. For a book so large, the index is fairly small, but the chapters are small and easily scannable. If you can find a spare couple of inches on your bookshelf, fill them with a copy as soon as you can. Computer Manuals, who distribute Abacus books in the UK, will ship you a copy if your local shop don't have any.

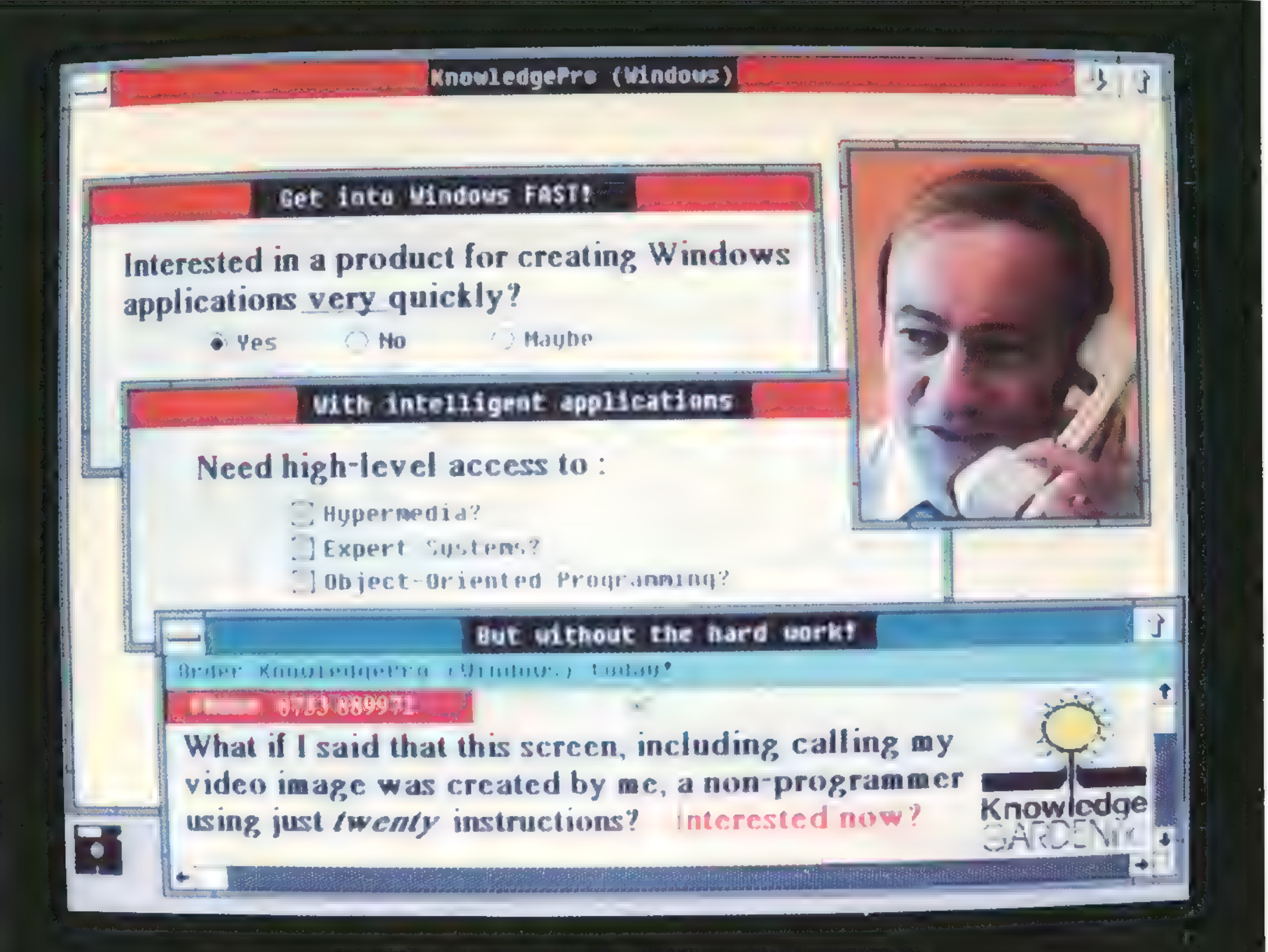
Author: Michael Tischer
Publisher: Abacus
Price: £55.45
ISBN: 1-55755-036-0
Pages: 930

FoxPro Made Easy

This is just one of the large number of FoxPro books that are beginning to fill the .EXE shelves. Unlike some, this is definitely not just a dBASE book that's been through a search-and-replace operation on a word processor. It explains how to use all of FoxPro's features, including the mouse and windows. Despite the title, this book is more for programmers than end-users, and includes a thorough summary of all FoxPro's commands and functions. A very good introduction, especially as the FoxPro manuals themselves don't attempt to teach the basics of programming.



Author: Edward Jones
Publisher: Osborne/McGraw-Hill
Price: £21.95
ISBN: 0-07-881609-2
Pages: 780



KnowledgePro (Windows)

Get into Windows FAST!

Interested in a product for creating Windows applications very quickly?

● Yes ○ No ○ Maybe

With intelligent applications

Need high-level access to :

□ Hypermedia?
□ Expert Systems?
□ Object-Oriented Programming?

But without the hard work!

Order KnowledgePro (Windows) today!

Phone 0753 889971

What if I said that this screen, including calling my video image was created by me, a non-programmer using just *twenty* instructions? Interested now?

Un-retouched VGA screen image. Special hardware required for motion video.

Introducing the door into Windows!

Easy access to Windows

KnowledgePro (Windows) contains high-level commands for manipulating screen objects, lists, text, fonts, rules, external files and bitmap images. DLL and DDE support lets you integrate your own C routines with KnowledgePro and link your application directly to Excel and other Windows programs.

At a price you can afford

KnowledgePro (Windows) costs £525 with no runtime fees for applications. KnowledgePro for DOS costs £375. The systems run on IBM PC, XT, AT and PS/2 compatible machines with 640K of memory and a hard disk. KnowledgePro (Windows) requires Microsoft windows 286 or 386 version 2.X or greater.

Call 0753 889972 (FAX 0753 883983) for more information or write to: Intellisoft (UK) Ltd, 77A Packhorse Road, Gerrards Cross, Bucks SL9 8PQ. KnowledgePro for DOS demo costs £25. Access, Mastercard, cheque and official purchase orders accepted. All prices excluding VAT and P&P.

Another intelligent tool from

Knowledge GARDEN Inc.

KnowledgePro is a registered trademark of Knowledge Garden. Excel is a trademark of Microsoft Corp. IBM, XT, AT and PS/2 are trademarks of International Business Machines Corp.

CIRCLE NO. 929

The definitive database for micros and micro-networks.

MAIN FEATURES

- Clear Concise English-like programming language.
- Rapid system development and easy maintenance.
- Relational and Hierarchical structures.
- Variable length records.
- Powerful facilities for reading and writing files from other systems - e.g. Pegasus, Lotus 123.
- Run-time licences available.

OS/2, PC-DOS, MS-DOS, XENIX,
CDOS, MS-NET, NOVELL ETC.

Microft Technology Limited
The Old Powerhouse, Kew Gardens Station
Kew, Surrey TW9 3PS.
Telephone: 081-948 8255

£560+VAT Single User.
£950+VAT Network Version up to 5 Users.
£100+VAT each subsequent User.

CIRCLE NO. 931

dBASE LANGUAGE FOR VAX, UNIX & XENIX

Recital

RECITAL, a comprehensive relational database management system and fourth generation development environment, is TOTALLY upwards compatible with Ashton Tate's dBase III PLUS. Now you can migrate your PC applications to your VAX, or a wide variety of UNIX and XENIX platforms—*without reprogramming!*

RECITAL lets you upload dBase applications and associated data DIRECTLY from your PC to your VAX or UNIX machine. And once it's installed you can maintain and modify it using the same FAMILIAR syntax and commands as on the PC. Operators trained in dBase will understand RECITAL *instantly.*

The potential is exciting. Not only is the whole catalogue of affordable dBase applications yours to use, but you can easily develop entirely new programs using RECITAL. And since there are thousands of programmers already trained in dBase, finding skilled people becomes much easier. Not only that, with RECITAL, end users can develop and maintain databases, generate reports, design screen forms, and much more . . . without writing *any program code at all.*

RECITAL is functionally rich

□ Upwards compatible with
  - ● dBase III Plus,
  - ● FoxBASE Plus
  - ● Clipper

□ Multi-user relational DBMS
□ Concurrent processing
□ Transparent access to existing RMS files
□ Powerful fourth-generation language
□ Complete menu driven end user interface
□ Powerful forms based tools
□ Comprehensive report writing language
  - ● screens
  - ● reports
  - ● labels

□ 2-way interface to 20/20 spreadsheet
□ Keyboard macros usable in any form or menus
□ Flexible prototyping tool
□ Active data dictionary with full referential integrity constraints

RECITAL *complements* other software products and helps you *release* their full power.

dBase and dBase III PLUS are trademarks of Ashton Tate. VAX, VMS and RMS are trademarks of Digital Equipment Corporation. 20/20 is a trademark of Access Technology. FoxBASE is a trademark of Fox Software Clipper is a trademark of Nantucket Corporation UNIX is a trademark of AT&T. Xenix is a registered trademark of Microsoft Corporation. Recital is a trademark of Recital Corporation.

For further information please contact:
RECITAL CORPORATION LTD
South Bank Technopark
90 London Road
London SE1 6LN
Tel: 01-401 2727
Fax 01-633 9617

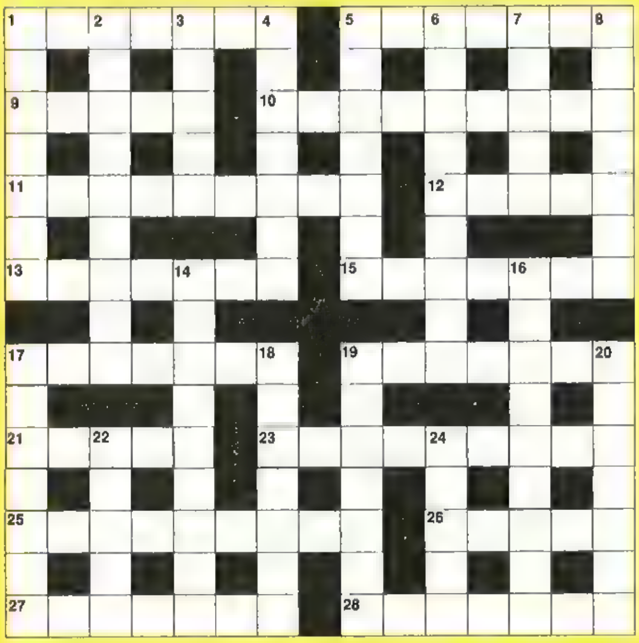
PROVE IT FOR YOURSELF!
Contact Recital Corporation today for details of our 30 day evaluation.

CIRCLE NO. 932

ADVERTISERS INDEX

ADVERTISER	CIRCLE	PAGE	ADVERTISER	CIRCLE	PAGE
2nd Computer Ltd	916	65	Penguin Software	922	51
3L	893	29	QA I	880	IFC
Borland	892	27	QA II	908	53
Calend	---	55	QA III	901	45
Central Point Software	896	35-37	Rainbow	883	7
Computer Bookshops	911	59	Richmond Systems	884	9
ControlTelemetry	904	49	Smalltalk Express	930	46
Data Encryption Servs.	910	57	Softlok I	903	49
Emap Exhibitions	890	23	Softlok II	921	70
Emtec	906	51	Software Con Co I	900	43
Fast Electronics	885	11	Software Con Co II	895	33
Glockenspiel	891	25	Software Con Co III	897	39
Grey Matter	881	3	Software Generation	888	19
Informix User Group	917	65	Software Ltd.	882	5
Intasoft	909	57	Software Security	926	OBC
Intellisoft	929	75	Sun Microsystems	898	41
IXI	907	70	System IX	899	42
Jensen &Partners	887	17	System Science	915	65
Link Computers	923	51	System IX	912	61
LPA	927	73	System Science	905	49
MSE	902	46	User Friendly	914	65
Microcosm	920	70	Verilog	886	14
Nantucket	889	20	Zortech I	918	67
Nu-Mega Technologies	893	30	Zortech II	919	69

.EXEWORD



ACROSS

1 Organise the pass, but not on time (7)
5 Measure of 17 dn in a channel (7)
9 Chunk of wood and a chip, computer basis (5)
10 I/O units often remote (9)
11 Plot a two-bit start in the machine (9)
12 Burning desire (5)
13 Chooses the chosen few somehow initially (7)
15 Working people on hills round French island (7)
17 Smart programming, structured neatly (7)
19 Brown male going off the edge (7)
21 Sounding huge, but may jar (5)
23 Aristocratic transport mechanisms? (9)
25 An entity's is in a field (9)
26 The first semi makes chunks of data... (5)
27 ... and long sets of characters (7)
28 Verses devilish for upsetting people (7)

DOWN

1 There often were eighty (7)
2 9, friend: you know it makes sense (9)
3 American code that's everywhere (5)
4 Pest ran wild and catches the bug (7)
5 Hill intended, I hear, to be agony (7)
6 On screen movement (9)
7 Sentinels that fly to warn us (5)
8 So, I scan the chaotic gaming places (7)
14 Centre of a condition (9)
16 Always keen on the environment like a Xmas tree (9)
17 Indications of data transfers (7)
18 Glitches in the gorge? (7)
19 Results of a successful search (7)
20 Somehow with a tic, Tess checks the chip (5,2)
22 Make changes at the church centre, I hear (5)
24 Amin to return a fool (5)

.EXEWORD APRIL

U	T	I	L	I	T	Y		D	I	S	O	W	N	S
P		N		N		A		E		C		R		E
D	E	C	K	S		W	I	N	D	O	W	I	N	G
A		R		E		N		O		U		T		M
T	H	E	I	R		I	N	T	E	R	F	E	R	E
E		M		T		N		E						N
S	I	E	V	I	N	G		S	I	G	N	B	I	T
		N		O						E		R		
L	A	T	E	N	C	Y		R	U	N	T	I	M	E
O						A		E		E		E		R
G	R	O	U	P	I	N	G	S		R	E	F	E	R
I		C		L		K		I		A		C		A
C	O	T		A		E		S		T	R	A	I	N
A		A		N		R		T		O		S		D
L	O	L	L	O	P	S		S	O	R	T	E	R	S

.EXE APPOINTMENTS

Group Recruitment Sales Manager
Jane Lyons 01-994 6477 x2238

Recruitment Sales Executives
Jeremy Cox 01-994 6477 x2337
Ed Butcher 01-994 6477 x2338

Process Communications Ltd, 10 Barley Mow Passage,
Chiswick, London W4 4PH Fax No: 01-994 1533

Rates:	1/4 page	132mm x 88mm	£520
	1/2 page	132mm x 179mm	£955
	1 page	273mm x 179mm	£1795

Copy Deadline: 18th of the month of publication.

Cancellation: Advertisements may not be cancelled after the copy deadline.

REAL TIME SOFTWARE ENGINEERS

All levels to Project Leader £15K — £24,000

Our clients include a number of leading software companies that are seeking to recruit high calibre personnel with a minimum of one years experience in development of real time embedded systems using ADA, PASCAL, 'C', OCCAM, UNIX, VAX/VMS and SUN.

Ranging from major manufacturers to specialised software houses, excellent opportunities exist to utilise your skills and to increase your experience in structured design techniques. All companies offer first class benefits in terms of salaries, career advancement and project responsibility.

If you have the experience that our clients are looking for contact us on 081 991 0932 or send your CV to:

(7149)

SÄNTIS
Electronics & Software
Recruitment Consultants

21 Rosebery Gardens
Ealing
London W13 0HD

Tel: 081-991 0932

PC SPECIALISTS - HAMPSHIRE/DORSET - to £22,000

We can offer you a choice of career opportunities with our clients, Hampshire & Dorset based companies who currently seek specialists with a minimum of 1 year's experience in any of the following;

- ✻ DATACOMMS
- ✻ UNIX and 'C'
- ✻ 4 GL's (ORACLE, INGRES etc.)
- ✻ CAD/CAM
- ✻ NETWORKING (esp. NOVELL, TOKEN RING)
- ✻ IMAGE PROCESSING

Excellent salary packages are offered, and in many cases generous relocation assistance is available...

For details of these and other career opportunities please write 'Freepost' quoting Ref: BS/PCS

CBS
APPOINTMENTS
BOURNEMOUTH BH1 1PW

Telephone: 0202-29255 Evenings & Weekends: 0202-681683

SOFTWARE ENGINEERS
TECHNICAL PROGRAMMERS
MATHEMATICAL MODELLERS

Vacancies are always available for graduates with at least a year's experience, and postgraduates, particularly if you have experience in any of the following languages.

FORTRAN ● ALGOL ● C ● CORAL
ADA ● PASCAL ● Assembly Languages

Send a detailed CV or call us now for an application form.

Our services are free to all applicants.

Ames Personnel, Ames House, King's Cross Lane,
South Nutfield, Redhill, Surrey RH1 5NG
Tel (0737) 82212 Fax (0737) 822133

AMES PERSONNEL

CLIPPER ANALYST – PROGRAMMERS

Intelligent, articulate analyst-programmers required by small central London software company set on expanding our successful PC banking system.

We program in Clipper backed by C and Assembler and need to produce a C version and port it to Unix and other operating systems. We use generators and software tools extensively and are implementing OOP techniques as they become available.

Science graduate preferred, with at least 2 years Clipper experience in networked banking or accounting applications. C, Unix or Assembler an advantage, curiosity and imagination an asset. Must be presentable and able to deal with clients when required. Initially, must be able to cope with supporting old products under pressure as well as working on exciting new projects from scratch. A mature attitude and understanding of business issues is essential.

The prospects are as good as you are and an exciting range of opportunities is available to the right person. Progression into either pure development or management is wide open.

Starting salary £18,000 to £20,000 depending on experience and ability.

We also have opportunities for trainees.

Please write enclosing CV to Trish McNamara at:

Synchronicity Ltd., Suite 103, City View House,
463 Bethnal Green Road, London E2 9QH

SOFTWARE – C/UNIX
UK , TO 25K

REAL TIME
Berkshire
16k

Software engineers required to carry out design, development and implementation of software or systems RT applications, 2-3 years experience ideally in a defence environment. Sound working knowledge of C, VAX-VMS and/or SUN UNIX operating systems is essential.

SIMULATION/MODELLING
Sussex

Intermediate and senior software engineers required, ideally with experience in Real Time Simulation using C and Assembler, to work on advanced simulators and training systems.

SENIOR SOFTWARE ENGINEERS
Berkshire
To 18k

Degree in Electronics, Computer Studies, Physics or related discipline with 2 years experience of embedded software design. A knowledge of high-level languages such as Ada, Pascal or C needed for design and implementation of Communications applications.

COMMUNICATIONS
Sussex

Software engineer required to develop communications interfaces to other manufacturers' products and systems and new networking protocols. Requires experience of at least one of the following:- Datacommunications protocols, Ethernet/Arcnet, C, Z80 Assembler or X25.

SYSTEM SUPPORT PROGRAMMER
Essex
To 20k

To provide technical support to a range of UNIX machines including VAX 500 and SUN 3 servers and workstations, working as part of a systems support team. Candidates should have a sound familiarity with UNIX support systems and an interest in a career in systems support.

PARALLEL PROCESSING – Hard/Software
West Country

Engineers required with good hardware and software skills at device and system level. Technologies include RISC processors, high speed ASIC's with device and system level software in C using a UNIX based operating system.

STS Recruitment

Send a CV in complete confidence or telephone.

33 Staple Gardens,
Winchester,
Hants SO23 8SR

Tel: 0962 69478
Eve/Weekends: 0962 760241
Fax: 0962 841982

Recruiting for Today's Technology

Given the enormous potential of technology throughout the 1980s, how can we accept the mere 8% growth in productivity? Steve Standring believes that it is time we addressed recruitment issues in a planned and systematic approach.

The centralised IT department is rapidly becoming obsolete. Within five years, I believe that 40-50% of the new jobs in IT will be in new roles. Today, the general trend appears to be moving more towards users and away from 'techies' and programmers. If this trend continues, then eventually the man who writes the requirement definition will also create the system - a scenario which implies that a radical change in values and orientations is required.

To get the most out of today's technology, new organisational structures have to be implemented. In fact there has to be a total rethinking of structures and skills based on future technology and business issues. We need to manage multiple organisation structures and different recruitment processes.

Not everyone is going to have the new skills for the jobs available, so rewarding people for jobs and skills is critical. Breaking the skills deficit cycle is going to require both more investment and a managed process.

Building and organising IT for the future requires a systematic approach. Initially we are seeing the need to assess an organisation's base preparedness to implement new technologies and to make change happen. We then need to design and plan for the IT structure of the future; the only way in which we can do this is by re-orienting the IT recruitment practices to implement these changes. The key to this is going to be the development and management of one's employees.

The new environment will require a wider perspective and a new emphasis away from technical skills and more towards business skills. Both IT and people working in IT will have to align themselves still further, ensuring that the business is run on a basis which is planned and understood by all participants.

In order to create complete cohesion between the IT staff and the business, we are going to see personnel/'human resource' management being dynamic, progressive, nurturing and supportive. This requires a non-hierarchical organisation and careful planning, of which there are five stages: Planning, Requirement Analysis, Design, Construction and Development/Maintenance. Only by concerted attention to the whole process can we place the correct candidate in a position appropriate to them, thus maximising efficiency and effectiveness.

Companies must no longer simply accept what is offered by external recruitment sources and must develop a relationship with a consultancy who feels part of the business solution, and who feels responsible for the business problem.

Steve Standring is Senior Executive of Computer People's Business Consultancy Division.

Connections

PERMANENT

LONDON

RECRUITMENT CONSULTANT - OTE £25,000 PLUS CAR

IT Connections are expanding and are seeking to recruit two more consultants for their W1 office. Ideally you will have at least six months IT recruitment experience preferably in a PC environment. Further training will be given where appropriate. We have excellent working conditions, regular advertising, and a custom built database. A good sense of humour is essential. For more information please contact Mike Oakes on 01 753-0160 or 01 487-6149 evening and W/Ends.

JUNIOR BUSINESS ANALYSTS - £16,000 + BENS

Minimum one years experience as an analyst preferably with PC skills. Excellent opportunity to progress in a 'greenfield' environment within the Insurance industry.

PC SUPPORT - WEST LONDON - £18-21,000 PLUS BANKING BENS

QS/2 Lan server experience essential. Very exciting and responsible position in State Of The Art environment.

2 X PC - DEVELOPERS - £16-22,000 - W. LONDON

Household name company requires dBase/Clipper skills in a Novell environment for new systems development. UNIX training available.

MICROVAX VMS PC SUPPORT - c£16,000 PLUS BENEFITS

This international bank based in the City are looking to recruit good VMS experience with a general knowledge of PCs. Training will be given in PC Software where appropriate.

MIDDLESEX

PC ANALYST - £15-20,000 + BENS

In the first instance this will be a support/evaluation role progressing later into a development role as well. You will be evaluating database products for future use. Novell, Excel and MS-Word useful but not essential

LAN SPECIALIST - TO c£19,000 + BENS

Large State Of The Art end-user requires Token-Ring LAN skills, SNA and OS/2 interest would be useful and you will be working on new State Of The Art systems development.

LAN/WAN SUPPORT - TO £17,000 + CAR

Networking specialist required for pre/post sales support and configuration. Novell and comms experience an advantage but not essential. Friendly atmosphere.

HANTS/DORSET

PC-DATABASE DEVELOPMENT - c£17-20,000 + BANKING BENEFITS

2 x PC-Analyst/Programmers required for new systems development. PARADOX, DBASE, FOXBASE, etc., experience required. Excellent opportunities in progressive bank.

3 X PC SUPPORT/DEVELOPMENT - TO £18,000 + BANK BENS

Bournemouth and Basingstoke available for PC Support and PC Development roles. Experience of Paradox, Foxbase and/or dBase/Clipper required.

LAN SUPPORT - BASINGSTOKE - TO c£12,000 + CAR

Experience of Novell and/or PC LAN required for total solutions company. Experience of Accountancy systems an advantage but not essential.

AVON/WILTS

2 X PC SUPPORT - c£16,000 + BENS

Swindon and Bath positions available with household name companies. Knowledge of Novell and Lotus 123 required. Large LAN/WAN installations so Connectivity training on offer.

BERKS

PC SYSTEMS (ACCOUNTS) ANALYST - c£15-18,000 + BANK BENS

Household names financial services company urgently requires people with Accounts background and PC experience eg. Lotus 123, Symphony etc. for support/development role.

PC SUPPORT - SLOUGH - TO c£16,000

Experience of Novell, Smart DW4 etc. required for this ideal 1st or 2nd jobber position. Good career move.

BUCKS

2 X PC-SUPPORT (DEV) - HIGH WYCOMBE AREA - c£14,000 + BENS

Large end-user urgently needs two support personnel at prestigious new offices. Experience of TOKEN RING/PC-LAN NETWORKS, LOTUS 123, DW4, DBASE and 3270 emulation experience an advantage, but not essential.

OXON

2 X PC - DEVELOPERS - c£16,000 + RELOCATION

Essential requirement for professionals at household name company to work with 'C' in a PC/Workstation and OS/2 R+D environment.

SUSSEX

PROJECT LEADER - SUB MORTGAGE - TO £21,000

Project leading skills essential for this position. You will be dealing with "EMERGING TECHNOLOGY" (Touchscreen Technology). Knowledge of C will be required, plus a good general knowledge of PC Hardware and Software.

MILTON KEYNES

PC DEVELOPMENT - TO £18,000

Good education plus dBase III + skills required for this consultancy. Very good corporate career growth.

CLIPPER PROGRAMMER - TO £18,000

Clipper programming skills required for Blue Chip company. Good opportunity to progress in this large I.T. Division

SURREY

3 COM EXPERT - c£22,000 + CAR

Excellent 3 Com Networking skills required for this specialist consultancy. Good promotional prospects envisaged ie. full consultancy.

SENIOR ANALYST PROGRAMMER FOXBASE OR DB3 - £21,000 + BENS

Excellent Foxbase or DBase skills required for this Insurance Company based in Surrey. Minimum 3 years experience programming. Analytical skills also essential.

BIRMINGHAM

PC SUPPORT - TO £15,000

Excellent opportunity for a person with at least 18 months experience in a support environment to join a consultancy committed to excellence. You must have skills in Lotus 123 (macro level) and DBase 3+ (Development Level) Good promotional prospects

HERTS

DEVELOPMENT + SUPPORT
NOVELL + CLIPPER - TO £16,000

CLIPPER + NOVELL skills essential for this number 2 position. Training given where necessary in other packages.

DBASE III + OR FOXBASE - TO £15,000

Knowledge of dBase III + or Foxbase required for a software solution company. Opportunity to progress quickly in challenging environment.

CLIENTS

IT CONNECTIONS is rapidly becoming the number one Recruitment Consultancy for PC Support and Development personnel. If you as a client are currently recruiting or expect to in the near future, then please call us now with your requirements in order that we may tailor our advertising to your needs.

I.T. CONNECTIONS
160 NEW BOND STREET
LONDON W1Y OHR
TELEPHONE: 071-753 0160
FAX: 071-753 0162

THAMES VALLEY

SOFTWARE/DESIGN ENGINEERS

SURREY £20,000

Our Client specialises in the design of electronic and computer systems, data acquisition data analysis, simulation and complex interfacing. They are expanding and seek graduate Engineers with the ability to write drivers, knowledge of screens and user friendly attractive software skills. Experience of 'C', UNIX, PASCAL, FORTRAN, OCCAM is essential.

ANALYST PROGRAMMERS

MIDDLESEX £18,000

Our client's central skills are the design and development of computer applications and digital and analogue electronics. Do you have a Computer Science or equivalent degree, with a minimum of 3 years experience in ADA, CORAL, PASCAL, C, YOURDON, MASCOT, JACKSON, within a real time environment. If you do they offer the opportunity to develop your career with one of the UK's leading engineering organisations.

HARDWARE/SOFTWARE ENGINEERS

SURREY £18,000

Our Client, who provides complete computer system solutions for defence and civil markets, currently seek highly motivated engineers to work on a variety of technically demanding projects. A degree in a scientific discipline with experience in any of the following is required: FORTRAN, CORAL, PASCAL, ADA, ASSEMBLER.

SYSTEMS ENGINEERS

SURREY £18,000

This small, rapidly expanding Computer Systems Design Company, specialising in graphics and image processing, seek candidates with experience of 'C', UNIX, OCCAM, PASCAL, ASSEMBLER to work on exciting projects utilising state of the art technologies. Candidates should possess a technical honours/good HND.

To find out more about these and many more exciting career opportunities please call LYNNE HOUGHTON on 0932 244014 (office hours) or 0784 247982 (eves/weekends).

THAMES VALLEY COMPUTER APPOINTMENTS
Franklin House
Station Approach
Shepperton
Middlesex TW17 8AN
TEL: (0932) 244014 (24 hours)
FAX: (0932) 253476

I.T. CAREER APPOINTMENTS

Project Manager
M4/M25 — to £27K+car
A strong 'man-manager' with extensive development skills in PC systems, LANS and PC/IBM M/F systems is req. to push on with extensive development of this international datacentre. Ref: EXE 02

PC Programmer
S/Counties — to £22K+bens
A solid background in applications development. PC development products and a desire to get involved in analysis are some of the key skills needed by this market leading IT systems house. Ref: EXE 03

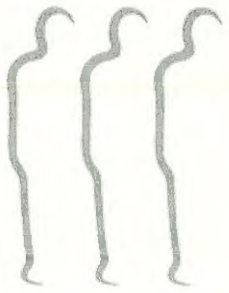
UNIX 4GLs
London/S.E. — to £23K
Min. 18 mths UNIX with exposure to COBOL, 'C' INFORMIX, ORACLE and CASE tools. S/W House ensures exciting projects. Ref: EXE 07

'C' A/Progs
London — to £20K
Min. 2 yrs with 'C' developing POS type systems. Client interaction is a key role in this appointment. Super benefits pkge. Ref: EXE 08

Software Engineer
Kent/Sussex — to £19K+
International group requires "client friendly" professionals with 'C' exp in PC/Windowing environment knowledge of real-time multi-tasking systems. Ref: EXE 09

VAX Programmers
S/Counties — to £20K
A number of clients have needs for Progs. with 12 mths min. exp. using ORACLE, INGRES, C and 4GLs. Wide spectrum of projects and career opportunities. Ref: EXE 10

For further information about these opportunities, or the many other client needs that we have in London and throughout the Southern Counties telephone Gill McKay – Resources Manager. Alternatively Post or Fax your resume to us.



PROJECT 3

I. T. RECRUITMENT SERVICES
LIMITED

Keeley House, 22-30 Keeley Road, Croydon CR0 1TE
Tel: 01 651 3216 (24 hrs) Fax: 01 651 3251

LONDON & SOUTHERN COUNTIES

PORTMAN OFFER A BETTER CHOICE IN ANYONE'S LANGUAGE

Unix Support — **to £25,000 + Car**
(Herts)
Expanding Systems house with global interests.
Key skills: Very good Unix with a knowledge of C from a product support background. Self starter, well presented and happy to travel internationally. A knowledge of Uniplex would be ideal.

'C' Analyst/Programmer — **£20 - 30,000 Package**
(City)
Leading International Banking Group
Key Skills: Minimum 3 years' experience in 'C'. Well educated with relevant financial experience, ideally Capital Markets. Development of front end dealing systems.

C/Unix Development — **to £20,000**
(Herts)
Established and expanding Systems House
Key Skills: 12 months' programming in C/Unix. Graduates preferred who are keen to develop skills in Windows, Graphics and Expert Systems.

Systems Developer/ Analyst — **£22,000 + Car**
(Middlesex)
Small and expanding consultancy, specialising in providing bespoke network solutions.
Key Skills: Systems analysis and development in an IBM PC and Apple Mac environment. Development tools such as dBase/Clipper, 'C'. Design of multi-user systems for a variety of clients.

Field Service Engineer — **£18,000 + Car**
(South London)
Expanding dealership.
Key skills: 18 months' field service experience. Ability to install, configure, plan and support Unix and Novell based systems. Experience of X25 and X400 would prove useful.

PC Support & Development — **£14,000 + Benefits**
(City)
Medium size firm of Chartered Accountants.
Key Skills: 1 year minimum in dBase III/Clipper, Lotus 1-2-3 and some graphics. A mixture of development and support. Age 20-25.

For further details, please contact, in confidence,
Mark Bamber on 01-236 1113 or 06284-3695 evenings/weekends.

PORTMAN RECRUITMENT COMPUTERS
15 GREAT SAINT THOMAS APOSTLE,
LONDON EC4V 2BB

PRC

Staff-Match

Recruitment Services Limited

UNIX/INFORMIX/SQL
Software Support
Surrey — Salary to £15,000
An excellent opportunity has arisen in the Software Support division of this leading international courier company. You will be involved in inhouse problem solving and supporting applications in the field. Experience in financial and accounting systems together with 12 months' UNIX/INFORMIX/SQL experience is desirable.

UNIX/INFORMIX/SQL
Post Sales Support
Middlesex — Salary to £20,000
This specialist Software House requires experienced self-motivated individuals to support Informix software. A minimum of 12 months' SQL experience is essential and UNIX desirable.

'C' Programmers
Surrey — Salaries to £13,000 + exc. benefits
Our client, a prestigious Assurance company, are currently recruiting 'C' programmers with as little as two months' experience. The successful candidates will also be trained in UNITY.

UNIX/C/GRAPHICS/WINDOWS
Software Engineers
Berkshire — Salaries to £21,000
This well established company, whose specialist area is image processing, are seeking well qualified, self-motivated individuals. You will be involved in all stages of a project life cycle, from requirement evaluation, systems design through to customer support.

PC/UNIX/AIX/BASIC
Programmers
Middlesex — Salaries to £15,000
Exciting opportunities exist within this major systems house who specialise in warehouse and point of sale systems. All that is required of you is to have 12 months' BASIC programming experience and have flair and initiative.

UNIX/C/SSADM/YOURDON
Senior Software Engineer
Hants — Salary to £22,000 + benefits
Our client, a high profile communications company, require a professional individual with design experience in a real-time/communications environment. The successful candidate will have a minimum of 2 years' design/programming experience using UNIX and 'C' and a structured methodology.

For further information, please telephone
Allison Barnard on 048649 595 or evenings
on 048649 780 or send your CV to
6 The Mews, The Common, Dunsfold, Surrey GU8 4LJ
or Fax to 048649 739

STOB - Don't give a XXXX

'The hacker...left a stream of abusive messages and hollow threats... I don't think KYLIE MINOGUE is his real name' - report of a hacking attack on the HALFWITS bulletin board.

The scene: Daytime, exterior in a domestic yard. At the back of the set is a small wooden toolshed. There is an extremely conspicuous telephone cable rigged up to go in through its window. A young man, with an outrageous footballer's haircut, enters the yard.

SCOTT: G'day. Anybody at home?

HENRY (out of vision): Who's that?

SCOTT: It's me, mate.

Henry appears from the house.

HENRY: G'day.

SCOTT: G'day. I wondered if you had seen Charlene today?

HENRY: I think that she's fooling around with her stamp collection in this little hut again. Why?

SCOTT: I just had a look at my bank balance, and somebody's just paid $25 million into it.

HENRY: That would be the third time this week.

SCOTT: Yeah. Call me suspicious if you like, but I'm beginning to think that something fishy is going on around here.

HENRY: And you think that cutie-Charlene has a hand in it? She's so nice, and clean, and wonderful.

SCOTT: Aw, don't say that, Henry. She's a good Sheila, and not at all stuck up.

(That's enough Python - Ed.)

Enter Jane, carrying a box which has MODEM printed on it in large letters.

HENRY & SCOTT: G'day.

SCOTT: What have you got in there, Jane?

JANE: Just some new stamps for Charlene, Mister Nosy.

SCOTT: All right, keep your hair on.

Jane goes into hut, closing door behind her.

JANE (o/v): G'day, Charlene. I've got the new modem.

CHARLENE (o/v): That's great, Jane. Now we will be able to hack into more systems, steal money and leave anonymous, abusive messages on bulletin boards.

JANE (o/v): Great idea, Charlene.

SCOTT: Hey, Charlene!

CHARLENE (Opens door): Hello Scott. There's something important I've got to say to you.

SCOTT: Henry, me and little Todd are off to help Ernie, the arthritic donkey, find a new home.

CHARLENE: It's about us, Scott. I now know that I could never marry a man without his own 386-compatible, and subscriptions to half a dozen of the more interesting bulletin board services.

SCOTT: Ernie has just been callously abandoned, to fend for himself, in our modern, growing city of Melbourne. Some people are just unbelievable.

CHARLENE: Our relationship is on the rocks, Scotch, sorry, Scott.

SCOTT: We are going to take some nice, juicy carrots. We wondered if you wanted to come along.

CHARLENE: Aw, you never listen to anything I say.

She runs off in tears. JAPANESE TOURIST walks onto set, brandishing map of the London Underground.

J/T: 'Scuse me. Please tell how to Cockfosters?

ENTIRE CAST: Drink it warm, mate!

J/T: Thank you velly much.

Exits. Jane walks out of hut, carrying a personal computer.

HENRY: What have you got there, Jane?

JANE: Just part of Charlene's stamp collection, Mister Nosy.

Exits.

SCOTT: 'Struth! Will you look at that! Call me suspicious if you like, but I think something's up with Jane and Charlene... *(and so on, and on, and on.)*

.EXE

IMAGE PROCESSING

Surrey ***to £17,000 + benefits***

Exciting opportunities exist within this rapidly expanding consultancy, with possible travel to USA, Far east and Europe..

People are needed with experience in Image Processing/ Recognition, 2D/3D graphics, UNIX, 'C' or Assembler to work on state of the art systems.

Excellent benefits include flexitime, company pension and life assurance schemes together with a stimulating working environment and rapid career progression.

EXR383/A

CAD SUPPORT

Cambridge ***£11,500 to £19,000***

We are currently a world leader and to strengthen our position in the CAD/CAM market, we are looking for engineers to join our team responsible for ensuring that we release the highest quality products.

You will have an engineering degree, together with some industrial experience and have a strong interest in advancing into CAD/CAM. Or, alternatively already have several years experience in a design or manufacturing environment using or supporting a CAD/CAM system.

A stimulating working environment awaits you, using the very latest VAXs, SUNs and PRIMEs in our newly equipped building.

EXR383/B

FIRST CAREER MOVE?

Hertfordshire ***to £19,000***

If you graduated at least 12 months ago and since then have gained solid 'C' experience, then this company would be extremely keen to talk to you about developing your career. They are probably the worlds leading systems developer of total office information and automation systems within the UNIX market place. They develop databases, graphical presentation systems using mixed colour full screen multi windows systems and much more.

EXR383/C

UNIX 'HIGH FLYERS'

Surrey ***to £25,000 + car***

This company must rank as one of today's most impressive organisations. Few companies can boast their level of achievment and success. In this role you will be working on the design of new CASE tools and datebase products within a structured environment. You will need to be of graduate calibre, with at least two years indepth 'C' and UNIX experience.

EXR383/D

SUN, 'C' & GRAPHICS

Sussex ***£14,000 to £21,000***

This company is renowned for its work in the area of advanced computer integrated manufacturing systems. Current projects revolve round real-time data acquisition, communications and database applications, although there is a strong emphasis on graphical user Interface work using networked SUN workstations. They are currently seeking HND degree level candidates with at least 12 months solid UNIX and 'C' experience.

EXR383/E

'C' & UNIX CAD CAM

North West ***to £25,000***

This major micro electronics design organisation is developing the latest state of the art VLSI chip design system. You should have excellent 'C' and UNIX experience gained in a workstation environment and ideally a knowledge of CAD development.

EXR383/F

'HIGH FLYERS' MOVE INTO CONSULTANCY

London/Manchester ***£20,000 +***

Whatever your IT background, these openings represent the very best career opportunities available to young, ambitious computer professionals (with partnership and extremely high earnings possible by your mid thirties). However, we only seek the 'creme de la creme'. Candidates must have an excellent academic record (1st or 2i University degree) plus three to six years IT experience. Presentation and communication skills must be of the highest order: mobility is also important. To really accelerate your career, call today.

EXR383/G

UNIX SUPPORT MANAGER

Surrey ***to £35,000 + BMW + benefits***

This company must rank as one of today's most impressive UNIX workstation manufacturers. In this role you will act as a regional support manager responsible for three support profit centres. You will also act as a technical focal point for their prestigious accounts. They are seeking graduate calibre candidates with a proven UNIX software project or support management background.

EXR383/H

RETRAIN IN X-WINDOWS

Berkshire ***to £16,000***

This company develop some of the most impressive multi-window, full colour terminal emulation systems for the UNIX market place. Utilising X-Windows and DEC Windows they seek enthusiastic individuals to train in these environments. You should have a good technical degree and a minimum of 12 months 'C' and UNIX knowledge. If you have solid DOS and/or MS-Windows and seek to retrain, this organisation will give you the opportunity.

EXR383/J

UNIX SUPPORT

Hertfordshire
£15,000 to £26,000 + car

This major UNIX systems house has enjoyed a spectacular growth and continues to be a most powerful force in the UNIX marketplace. You will perfrom a highly technical post sales and consultative role for their diverse range of UNIX software tools. Of graduate calibre, you will need at least 18 months solid UNIX and 'C' experience. You may have gained your experience within a UNIX development or support environment, but above all you must have excellent interpersonal skills.

EXR383/K

X-WINDOWS ARCHITECT

Thames Valley ***to £25,000***

Regarded as one of today's most innovative and exciting development centres in Europe, you will be providing design and consultancy expertise for this company's collage Windowing system. You will act as a design authority utilising X-Windows and OSF/Motif. Strong contenders for this role will be degree qualified, together with at least three years UNIX and 'C' and around ten months or more X-Windows experience.

EXR383/L

SECURITY CONSULTANTS

Hampshire
all levels £13,500 to £30,000 + benefits

A computer systems security policy comprises three elements: communications security; computer security; and operational security.

This unique software company addresses all three areas and is believed to be Europe's largest specialist team with indepth knowledge of computer and communications security.

With a largely untapped marketplace, its future success looks guaranteed and in order to keep abreast of continued demand for their services, they now require computer professionals with the following areas of expertise:

'C' and UNIX ★ Assembler (ideally M68000) ★ Low level hardware and firmware knowledge ★ 4GL and SQL experience ★ Project Managers Workbench ★ COMPUSEL knowledge ★ Secure Operating Systems ★ Gould ★ Formal methods ★ INGRES RDBMS Design & Prog.

EXR383/M

VIDEO GRAPHICS

Thames Valley ***£14,000 to £22,000***

In this role you will be engaged on the software design of digital graphics products including video and animation software and image processing primitives. You will need at least 18 months postgraduate experience in writing real-time software (in 'C' or Pascal) to drive hardware systems.

EXR383/N

'C' COMPILERS

Thames Valley ***£14,000 to £19,000***

This small but highly acclaimed development group are engaged in the design of 'C' translators and compilers for use on ultra fast RISC based workstations. This position would certainly suit a graduate calibre software engineer with at least twelve months UNIX development experience, seeking a truly advanced research environment.

EXR383/P

OPEN SYSTEMS CONSULTANTS

London ***to £35,000 + car + benefits***

This company must rank as one of the top 'City' consultancies. They are currently seeking degree level candidates with at least five years in-depth experience in one of the following areas: UNIX and networking, OSI COMMS, RDBMS. As a Principal Consultant you will conduct high level technical assignments within major blue chip organisations.

EXR383/R

Name (Mr. Miss. Ms. Mrs.) ____________________

Address ____________________

____________________ Post Code ____________________

Home Tel: ____________ Day Tel: ____________ Ext: ________
(Discretion assured)

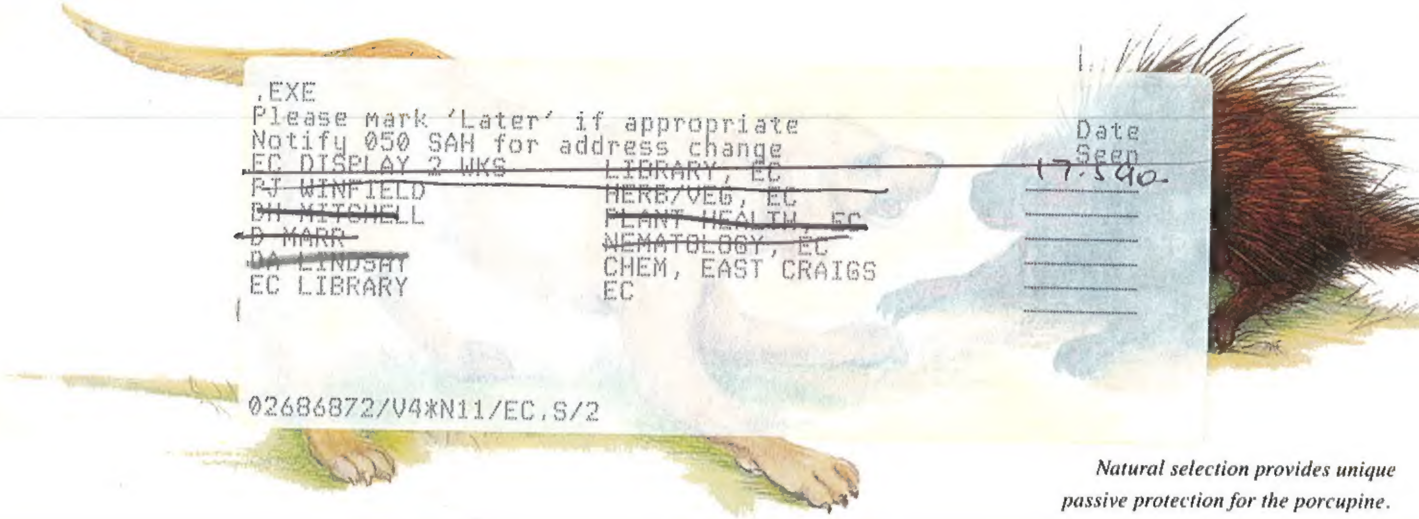
General enquiry only – tick box □ I have enclosed my C.V – tick box □ Advert Ref No: **EXR383**

Modus Recruitment
The Tythe Barn,
High Street, Edlesborough,
Dunstable, Beds. LU6 2HS

MODUS

☎ 0525 222222

Available for MS-DOS and Macintosh



Natural selection provides unique passive protection for the porcupine.

The Activator - Natural Selection For Software Protection

Inventor and entrepreneur Dick Erett explains how "The Activator" provides sane protection for your intellectual property.

"In any industry, just as in nature, the process of natural selection raises one solution above another. Natural selection is the most elegant of engineers.

In the area of software protection The Block has been selected by the marketplace as the solution that works. Over 500,000 packages are protected by our device.

For the past 4 years our philosophy has been; *'You have the right and obligation to protect your intellectual property.'*

A New Ethic For Software Protection

In allowing end-users unlimited copies of a software package and uninhibited hard disk and LAN operation, The Block has created a new ethic for software protection.

By removing protection from the magnetic media we remove the constraints that have plagued legitimate users.

They simply attach our key to the parallel port and forget it. It is totally transparent, but the software will not run without it.

A New Technology For Software Protection

Our newest model, The Activator, builds on our current patented design, and establishes an unprecedented class of software protection.

We have migrated and enhanced the circuitry of The Block to an ASIC (Application-Specific Integrated Circuit) imbedded in The Activator.

This greatly improves speed and performance, while reducing overall size. Data protection can also be provided.

Programmable Option

The Activator allows the software **developer** the option to program serial numbers, versions, or other pertinent data known only to the developer, into the circuit, and access it from the program.

Once you program your part of the chip, even we have no way to access your information.

The ASIC makes emulation of the device virtually impossible. It also presents an astronomical number of access combinations.

Full 100% Disclosure

Since The Activator is protected by our patent we fully disclose how it works. Once you understand it, endless methods of protection become evident.

Just as no two snowflakes are the same, no two implementations of The Activator are identical. And like the snowflake the simplicity of The Activator is its greatest beauty.

We never cramp your programming style or ingenuity. Make it as simple or complicated as you desire.

Let us help safeguard what's rightfully yours. Please call today for additional information or a demo unit. *It's only natural to protect your software.*"

Software Security UK LTD
21A The Precinct, High Street, Egham
Surrey TW20 9PN
TEL: (0784) 430060
FAX: (0784) 430050

"On-the-fly" programmable memory option now available for OS/2, MS-DOS and Macintosh

Unlimited Copies • Programmable • No Batteries • Small Size • Fast • Patented • Data Protection

CIRCLE NO. 926